# MAXIMUM AND MINIMUM ENTROPY METHODS OF IMAGE RESTORATION

A thesis submitted
in Partial Fulfilment of the Requirements
for the degree of
MASTER OF TECHNOLOGY

by

CAPT. P. K. JAGGIA

to the

DEPARTMENT OF ELECTRICAL ENGINEERING
INDIAN INSTITUTE OF TECHNOLOGY, KANPUR
AUGUST 1984

1/8/84

# CERTIFICATE

Certified that the thesis entitled 'MAXIMUM AND MINIMUM ENTROPY METHODS FOR IMAGE RESTORATION' by Capt. P.K. JAGGIA has been carried out under my supervision and this has not been submitted elsewhere for a degree.

S.K. Mullick

Aug. 1,1984

( Dr. S.K. Mullick )
Professor
Department of Electrical Engineering
Indian Institute of Technology
Kanpur

EE-1984-M-JAG-MAX

ACKNOWLEDGEMENT

I wish to express my sincere thanks to my thesis supervisor Dr. S.K. Mullick for his wise counsel and encouragement at various stages of this thesis. The literature provided by him and the useful discussions I had with him has helped immensely in the completion of this thesis.

I am thankful to Dr. M. Rama Mohan Rao for the useful discussions I had with him on certain mathematical aspects.

Capt. P.K. Jaggia

## ABSTRACT

Recent interest has centered on nonlinear techniques for the restoration of images. These techniques improve the resolution of blurred images without introducing erroneous details. Maximum Entropy method formulations incorporate these features and has been described and simulated on DEC-1090 computer system. The iterative nature of the solution requires extensive use of the computer system. Restoration results are obtained for artificially blurred arbitrary image data.

Minimum Entropy Deconvolution for restoration of images is another technique which is described and simulated for a class of signals which have essentially a non-Gaussian probability distribution and are impulsive in nature.

The description of various techniques and the software developed for their implementation are described in elaborate details.

TABLE OF CONTENTS


| | | Page |
|---|---|---|
| CHAPTER 1 | INTRODUCTION | 1 |
| CHAPTER 2 | MAXIMUM ENTROPY RESTORATION FRIEDEN'S APPROACH | |
| | 2.1 Maximum Entropy Principle | 10 |
| | 2.2 Mathematical Formulation | 11 |
| | 2.3 Frieden's Maximum Entropy Solution | 13 |
| CHAPTER 3 | MAXIMUM ENTROPY RESTORATION ONE DIMENSIONAL SEARCH ALGORITHM | |
| | 3.1 Introduction | 19 |
| | 3.2 Mathematical Formulation | 19 |
| | 3.3 Solution of Differential Equations | 31 |
| | 3.4 Development of the Algorithm | 38 |
| | 3.5 A Powerful Maximum Entropy Method | 42 |
| CHAPTER 4 | MINIMUM ENTROPY DECONVOLUTION | |
| | 4.1 Introduction | 47 |
| | 4.2 Minimum Entropy Principle | 47 |
| | 4.3 Objective Functions | 50 |
| | 4.4 Partial Order $\succeq$ | 52 |
| | 4.5 Uniqueness of the Model | 56 |
| | 4.6 Characterization of Admissible Objectives | 57 |
| | 4.7 Image Restoration Problem | 60 |
| | 4.8 Solution | 62 |

Page

CHAPTER 5 SIMULATION AND RESULTS

5.1 Frieden's Maximum Entropy Restoration 64

5.2 One Dimensional Search Algorithm for Maximum Entropy Restoration 70

5.3 Minimum Entropy Deconvolution 72

CHAPTER 6 CONCLUSIONS 75

REFERENCES

APPENDIX A

APPENDIX B

# CHAPTER 1

# INTRODUCTION

The basic purpose of collecting data is to extract meaningful information about the phenomenon of interest. Often, the phenomenon is not a direct physical observable and the information about it is only obtainable through some measuring instrument or system. The data available may be a linear superposition of the desired quantities. This type of linear information mixing is peculiar to a large number of diversified fields in physical sciences. The main problem confronted in these fields is how to unmix the data. When the phenomenon of interest is an image, the process of unmixing the data is termed as 'Image Restoration'.

Image restoration deals with images that have been recorded in the presence of one or more sources of degradation. There are many sources of degradation in imaging systems. Some types of degradation affect only the gray levels of the individual image points, without introducing spatial blur. Such degradations are called point degradations. Other types, which involve blur, are called spatial degradations.

For the purposes of digital simulation, based on the convolutional model, only spatial blur degradations will be considered.

In optical image formation, an unknown spatial radiance distribution o(x) called 'The Object' produces an image irradiance distribution i(y) which, for the purpose of digital processing, is collected as data $i(y_m)$, m=1,...M (or for simplicity of notation as i(m), m=1,...M). It is desired to estimate o(x) at a discrete subdivision of points $x_n$, n = 1,...N. The relation between the image and the object, in one dimensional notation, is the linear form

$$i(y_m) = \int_{-X}^{X} o(x)h(y_m,x)dx + v(y_m), \quad m=1,\dots M \qquad (1.1)$$

where h(y,x) designates the point spread function of the overall image forming system and v denotes additive noise, random or otherwise. The assumption that noise is additive is subject to critism. However, the assumption of the additivity of noise makes the problem mathematically tractable. The finite limit X denotes the finite extent of the imaging system, i.e. the image forming system covers only a finite region from -X to X, such that

$$h(y,x) = 0 \quad \text{for all } |x| > X \qquad (1.2)$$

The main interest is to estimate or restore o(x) only over this region. It has been firmly established that the ability to restore an object image o(x), from the observation of its

degraded image $i(y_m)$, is ultimately limited by the noise $v(y_m)$ in the acquired image data. More specifically, the ratio of signal to noise matters, and the mean squared restoration error is found to vary in an inverse manner with the signal to noise ratio.

To facilitate finding $o(x)$ at the discrete subdivisionsx $x_n$, $n = 1,\ldots,N$, it is convenient to replace the integral in (1.1) by an approximating sum, by numerical quarature, such that

$$i(y_m) = \sum_{n=1}^{N} \omega_n \, o(x_n) \, h \, (y_m, x_n) + v(y_m), \quad m = 1,\ldots,M \tag{1.3}$$

Numbers $\omega_n$ are input weights for facilitating accuracy in the approximating sum. If, except for translation, the degraded image of a point is independent of the position of the point, then the point spread function takes the form $h(y-x)$ and eq. (1.1) becomes

$$i(y_m) = \int_{-X}^{X} o(x)h(y_m-x_n)dx + v(y_m), \quad m=1,\ldots M \tag{1.4}$$

In this case the degradation is termed shift invariant. The assumption of shift invariance shall be considered throughout this thesis.

From mathematical stand point, given the model (1.4) and the degraded image $i(y_m)$, the aim of restoration is to

make as good an estimate as possible of the original image $o(x_n)$. Evidently, any such estimation procedure must require some form of knowledge concerning the degradation function $h(y,x)$ in (1.4). This a-priori knowledge can be effectively used to place constraints on the solution of the restoration algorithm. One such constraint is non-negativity of the restored pixels of the object image.

There is no single optimum method for estimating a general $o(x)$. There is a best restoring method relative to the type of a-priori information regarding object and the noise one might have at hand. Such types of prior information about object shape or object and noise statistic are important in fostering a choice of the restoration method. Other factors, such as the amount of data to be processed and the computing cost also influence the choice.

The various restoration techniques can be devided under two main categories

a) Linear methods

b) Non-linear methods

The linear methods consist mainly of techniques of inverting the discrete imaging equation (1.4) or linearly inverting or filtering the continuous version (1.1). These methods can be analysed by analytical means and are

widely used due to their computational speed. In contrast, the non-linear methods are difficult to analyse and are heavy in computational cost. Very often their implementation requires iterative schemes of computation. Thus, the basic question that arises is why use the non-linear methods. The fact underlining the efficiency of non-linear methods is that for the optical signal images the linear methods are not optimum because these images in general do not follow normal statistics, being always positive quantities. Also, the resolution of a blurred image of an object can be improved by various linear techniques, but oscillations or ringing around sharp changes of intensity in the image often induce erroneous details at the same time. What is required is a safe inversion technique, giving restorations free from ringing so that confidence can be placed in the reality of all the extra details revealed by the restoration method. Recent interest has therefore centred on non-linear techniques which incorporate constraints to reduce the artifacts generated in the restoration.

Maximum entropy restoration falls under the category of non-linear methods. It is a fundamental method for the solution of the inversion problem in image restoration. Stated most briefly, it selects the probability distribution that has maximum entropy permitted by the information we have

The reason for the use of maximum entropy criterion for restoration is as follows. In constrained deconvolution first a roughness measure is constructed which is then minimized subject to some constraints. Clearly many other roughness measures are possible, whose minimization or maximization would cause a smoothing influence on the restored images. For example, consider the function

$$H = - \Sigma \; x \ln x \quad , \tag{1.5}$$

where $x \ln x$ will always exist since the image gray levels are essentially non-negative numbers. Structure of (1.5) is similar to that of conventional entropy measure which is defined for a discrete random process and gives its information rate. Here the entropy is defined for a single deterministic non-negative function and serves as its roughness measure. Maximization of this entropy measure causes a smoothing influence on the restoration. This smoothing influence is both on the noise as well as the object image. Thus a trade-off is achieved between the contrast of the object image and the effect of the noise. In certain algorithms noise can be smoothened out more than the object image, thus producing better restorations. Also, due to the logarithmic nature of the entropy measure, its solution is deemed to be positive and hence the positivity constraint

is taken care-off inherently. This aspect scores over the linear methods of the restoration. Furthermore, the maximum entropy solution results in the least biased estimates and protects the restoration against spurious details for which there is no evidence in the observed data.

Chapter 2 of this thesis discusses one such maximum entropy technique for the restoration of images. This is based on the algorithm developed by B.R. Frieden [1,2] and is essentially a (M+1) dimensional search problem, where M is the number of pixels being restored. Due to the large dimension of the problem as well as the iterative nature of the scheme for obtaining the solution, the computational time requirements are very large for images depicted by larger arrays.

Chapter 3 discusses another maximum entropy technique wherein the search is directed along one dimension only for obtaining the desired solution for the restoration. Though this method is also iterative, but being a one dimensional search problem, it requires much lesser computational time for larger arrays as compared to (M+1) dimensional search algorithm.

In contrast to the principle of maximum entropy restoration is the concept of 'minimum entropy deconvolution'

This method is widely used in the field of seismology, where for a given time series y, which is a filtered version of white noise x such that

$$y = f * x \quad , \tag{1.6}$$

the deconvolution problem is to find a filter b which deconvolves or recovers the series x from the observed series y, i.e.

$$x = b * y \tag{1.7}$$

This technique can be adapted for the purpose of restoration of a class of image signals which are impulsive and essentially have a non-Gaussian probability distribution. The underlying principle, on which this method is based, is that the minimization of the entropy of the estimated image data restores the contrast or order in the estimated image. This is so because a data with non-Gaussian probability distribution has much order which can be restored by ensuring the solution to be as much away from the Gaussian distribution, as much permitted by the observed data. Thus, we require defining a norm that measures the order in the desired signal data. Another feature of this method is that the a-priori information about the point spread function of the degrading imaging system is not needed at all. Thus, this technique

can be literally termed as 'Minimum Entropy Blind Deconvolution'. Chapter 4 discusses all about this technique.

The techniques discussed in Chapters 2,3 and 4 have been simulated on the computer DEC 1090 system. The computer implementation and the results obtained thereof are discussed in Chapter 5. This is followed by the concluding remarks in Chapter 6. The characterization of the digital image and the finite area superposition operator is discussed at Appendix A. The computer program listings as well as the simulation results obtained are attached as Appendix B.

# CHAPTER 2

## MAXIMUM ENTROPY RESTORATION: FRIEDEN'S APPROACH

### 2.1 MAXIMUM ENTROPY PRINCIPLE:

Entropy of a continuous probability density function $p(x)$ is defined as

$$H = -\int_{-\infty}^{\infty} p(x) \ln p(x)\, dx \qquad (2.1)$$

and for the discrete case as,

$$H = -\sum_{i=1}^{N} p_i \log p_i \qquad (2.2)$$

The rationale for using the principle of maximum entropy is as follows. A problem similar to the inversion of the basic imaging eq. (1.1) is the inversion of

$$q(m) = \int_{-\infty}^{\infty} dx\, s(m)\, p(x), \quad m=1,2,\ldots M \qquad (2.3)$$

for the unknown probability density $p(x)$ given M of its moments $q(m)$, which corresponds to the input image data $i(y_m)$, $p(x)$ corresponds to $o(x)$ and must be positive to represent a real probability density, and $s(m)$ is some weighting factors. For this probability-estimation problem, it has been shown that the 'least-biased' estimate $p(x)$ is the unique function which has a maximum entropy

$$H = -\int_{-\infty}^{\infty} dx\ p(x)\ \ln\ p(x) \qquad (2.4)$$

subject to the data eqn. (2.3). Least biased means as equiprobable as the input moments will allow. Thus, the maximum entropy condition exerts a smoothing influence on the estimates, and ensures an all-positive output.

## 2.2 MATHEMATICAL FORMULATION

Measures of entropy used in image restoration are many, and have been widely used in different image processing schemes. Frieden's entropy measure [1,2] is discussed here.

The statistical model for the object is assumed to be composed of discrete, mathematical grains of small intensity $\Delta(o)$ which are distributed over the object scene. The scene is sub-divided into cells centered on a subdivision of points $x_n$, and the unknown object is assumed to have $O_n$ grains in cell n. Thus, recalling the one dimensional imaging eqn. (1.2),

$$o(x_n) = O_n \Delta_o;\ n=1,\ldots N \qquad (2.5)$$

Let $p_n$ represent the probability of a grain locating in cell n. If a large number of grains are distributed over the object, then by the law of large numbers

$$p_n = O_n/O_T,$$

where $O_T$ is the total number of grains in the object. $O_T$ is assumed to be known from the image data by the law of conservation of energy.

The entropy, then, is

$$H = -\sum_{n=1}^{N} p_n \ln p_n = \sum_{n} (O_n/O_T)\ln(O_n/O_T)$$

Substituting $O_n$ from eq. (2.5), it follows

$$H = -\sum_{n} (o(x_n)/\Delta_o O_T)\ln (O_n/\Delta_o O_T) \qquad (2.6)$$

By the definition of $O_T$, we finally get,

$$H = -\alpha \sum_{n} o(x_n)\ln o(x_n) + \beta$$

where $\alpha,\beta$ are constants.

The principle of maximum entropy then becomes

$$H = -\sum_{n} \hat{o}(x_n)\ln \hat{o}(x_n) = \text{maximum} \qquad (2.7)$$

where $\hat{o}(x_n)$ is the estimate of $o(x_n)$.

The Frieden's entropy measures, developed above, is essentially a concave function. Thus it has a unique maxima and hence a unique solution. The absolute maximum H in fact results from the perfectly smooth estimate

$\hat{o}(x)$ = constant. The input image data, however, acts as constraints which force fluctuations in $o(x)$ and hence departure from the absolute maximum H situation.

## 2.3 FRIEDEN'S MAXIMUM ENTROPY SOLUTION:

In this approach, a weighted sum of the image entropy and the observation noise entropy is maximized. An unbiased or maximum entropy estimate of noise is also desired now. However, the noise in the image data can be both positive and negative, and the logarithm of a negative quantity is undefined. This difficulty can be overcome by applying additional a-priori knowledge into the problem and assuming that a constant B is known such that

$$B = - \text{most negative } v(y_m) \tag{2.8}$$

where $v(y_m)$ is the observation noise at point m, in eq.(1.1). Thus, we define a new set of noise values

$$V(m) = v(m) + B, \qquad B \geq 0 \tag{2.9}$$

B is a positive constant of sufficient size that it cancels the most negative going values v(m) and thereby makes all $V(m) > 0$. But precise knowledge of this number B is not available. However, if a-priori information about $\sigma$, the standard deviation of noise, is at hand then we can set $B = -2\sigma$.

The quality of the restorations do not critically depend on the use of a value B precisely satisfying eq. (2.8), however the solutions are best when eq. (2.8) is indeed satisfied.

Since the object and the noise are independent arrays of numbers, the total entropy from both is the sum of the two entropies. The input data which constrains the maximum entropy solutions away from perfectly flat estimates is the image data and its total energy $I_o$. The $I_o$ is equated to the total object flux by conservation of energy. In the digital representation of the image, $I_o$ implies the sum of the gray levels. The restoring algorithm, for one dimensional images, then becomes

$$-\sum_{n=1}^{N} \hat{o}(x_n) \ln \hat{o}(x_n) - \rho \sum_{m=1}^{M} \hat{V}(y_m) \ln \hat{V}(y_m)$$

$$- \sum_{m=1}^{M} \lambda(m) \left[ \sum_{n=1}^{N} \hat{o}(x_n) h(y_m - x_n) + \hat{V}(y_m) - B - i(y_m) \right]$$

$$- \lambda(M+1) \left( \sum_{n=1}^{N} \hat{o}(x_n) - I_o \right) = \text{maximum} \qquad (2.10)$$

The new input parameter $\rho$ allows for affecting varying degree of smoothness on $\{\hat{o}(x_n)\}$ and $\{\hat{V}(y_m)\}$. The larger is the value of $\rho$, the more emphasis is placed upon maximizing the entropy of the $\{\hat{V}(y_m)\}$, and hence upon

making it smoother. The unknown $\lambda_m$ , m=1,...M+1 are the Lagranges multipliers.

For the two dimensional images the single summations are replaced by double summations and the single indices by double indices i.e.

$$\sum_{m=1}^{N}\sum_{n=1}^{N} \hat{o}(m,n) \ln \hat{o}(m,n) - \sum_{m=1}^{M}\sum_{n=1}^{M} \hat{V}(m,n) \ln \hat{V}(m,n)$$

$$-\sum_{p=1}^{M}\sum_{q=1}^{M} \lambda(p,q) \sum_{m=1}^{N}\sum_{n=1}^{N} \hat{o}(m,n)h(p-m,q-n)+\hat{V}(m,n)-B-i(p,q)$$

$$-\lambda(M^2+1)\left(\sum_{m=1}^{N}\sum_{n=1}^{N} \hat{o}(m,n) - I_o\right) = \text{maximum} \qquad (2.11)$$

If the object and the image data are represented in vector-space form, then the two dimensional algorithm gets converted into the form (2.10) with numbers N and M replaced by numbers $N^2$ and $M^2$ respectively, assuming that object and image data matrices were square. Thus, for analytical purposes, the objective function (2.10) will only be considered.

The solution to (2.10) is obtained by differentiating it with respect to $\hat{o}(x_n)$ and $\hat{V}(y_m)$ over all m,n and equating the result to zero. Differentiating (2.10) with

respect to $\hat{o}(x_n)$ yields

$$-\ln \hat{o}(x_n) - 1 - \sum_{m=1}^{M} \lambda(m)\, h(y_m - x_n) - \lambda(M+1) = 0$$

or,

$$\hat{o}(x_n) = \exp\left[-1 - \lambda(M+1) - \sum_{m=1}^{M} \lambda(m) h(y_m - x_n)\right] \qquad (2.12)$$

Differentiating (2.10) w.r.t. $\hat{V}(m)$, we get

$$-\rho \ln \hat{V}(y_m) - \rho - \lambda(m) = 0$$

or

$$\hat{V}(y_m) = \exp\left(-1 - \lambda(m)/\rho\right) \qquad (2.13)$$

The unknown lagrange multipliers $\{\lambda(m)\}$ in the above solution are found by making the solutions (2.12) and (2.13) consistent with the input constraint equations

$$\sum_{n=1}^{N} \hat{o}(x_n) h(y_m - x_n) + \hat{V}(y_m) - B = i(y_m); \quad m = 1, 2, \ldots M \qquad (2.14)$$

and,

$$\sum_{n=1}^{N} \hat{o}(x_n) = I_o \qquad (2.15)$$

The right-hand sides of (2.14) and (2.15) are the input data. Substituting (2.12) and (2.13) in (2.14) yields

$$\sum_{n=1}^{N} \exp\left[-1 - \lambda(M+1) - \sum_{\ell=1}^{M} \lambda(\ell) h(y_\ell, x_n)\right] h(y_m, x_n) +$$

$$\exp\left[-1 - \lambda(m)/\rho\right] - B = i(y_m) \qquad (2.16)$$

Eq. (2.16) represents a system of M non-linear equations in M+1 unknown $\{\lambda(m)\}$. Along with eqn. (2.15), it forms a system of M+1 equations, that are non-linear in M+1 unknown $\{\lambda(m)\}$, This system of equations can be solved to produce the desired restoration.

There are many ways of finding the solution to the set of non-linear equations represented by (2.15) and (2.16). However, Newton-Raphson iterative scheme has been found to work effectively. In this, an initial set of $\{\lambda_m\}$ is chosen for m = 1,2,...M and $\lambda(M+1)$ is chosen so as to satisfy (M+1)th constraint equation (2.15). Usually $\lambda(m)$ is chosen = 0 for m=1,...M. The $\{\lambda_m\}$ are then changed in order to satisfy all the constraint equations. After 8 to 40 iterations of this algorithm, a set of $\{\lambda(m)\}$ is found satisfying (2.15) and (2.16).

It can be seen that the functional form of representation (2.12) of the object guarantees a positive estimate. Regarding spurious oscillations in the output O(x), we see from (2.12) that

$$d^k \hat{o}(x)/dx^k \quad \alpha \quad \hat{o}(x), \qquad k = 1,2,... \tag{2.17}$$

This implies that $\hat{o}(x)$ cannot oscillate where it is at zero background and can oscillate little where $\hat{o}(x)$ is small. Hence, it is very smooth over these regions and

completely lacking in spurious details. Due to this very reason, this method works quite well on objects consisting of multiple impulses, for here all but a finite number of object values are zero. The physical objects consisting of rectangular pulses of random spacing and intensity are line spectra and star fields.

The partial derivatives of $\hat{o}(x)$ with respect to $\lambda_m$, $m = 1,\ldots,M+1$, are also proportional to $\hat{o}(x)$, i.e.

$$\frac{d^n \hat{o}(x)}{d_1 \cdots d_{M+1}} \alpha \hat{o}(x) \qquad (2.18)$$

Therefore, where $\hat{o}(x)$ approximates zero it will be nearly invariant with changes to the solution set $\{\lambda_m\}$. Hence, if the same object is restored many times by use of differing sets $i(y_m)$ of noise-prone data, the solution sets $\{\lambda_m\}$ for these restorations will all be nearly invariant where $\hat{o}(x) \backsimeq o$. This amounts to a condition of stability to inputs where $\hat{o}(x) \backsimeq o$, and hence is a very useful property.

Further, the examination of object restoration eq.(2.12) reveals that each image input $i(y_m)$ contributes a degree of freedom to $\hat{o}(x_n)$. As the number of input image data decreases, $\hat{o}(x)$ approaches an increasingly smooth estimate. Conversely, larger M, freer is $\hat{o}(x)$ to fluctuate.

# CHAPTER 3

## MAXIMUM ENTROPY RESTORATION 'ONE DIMENSIONAL SEARCH'

### 3.1 INTRODUCTION:

This chapter deals with an algorithm which involves solving a system of ordinary differential equations with appropriate initial conditions for the solution of maximum entropy restoration problem. This algorithm, developed by Zhung Yu and Haralick [17], does not make use of any conventional optimization method. The main feature of this scheme is,that, it avoids searching in a (N+1) dimensional space as required by most maximum entropy algorithms, where N represents the number of pixels to be reconstructed in the desired image. This algorithm involves searching along a path in the (N+1) dimensional space which is defined by a differential equation system with appropriate initial condition which can be computed easily. Hence, this algorithm is essentially a one-dimensional search problem as compared to (N+1)-dimensional search problem in most ME algorithms.

### 3.2 MATHEMATICAL FORMULATION:

Let the underlying image be denoted by a set of positive numbers $f_1 \ldots f_n$. Defining the entropy measure on this set of numbers we get,

$$H(P_1 \ldots P_N) = - \sum_{i=1}^{N} P_i \log P_i \qquad (3.1)$$

where $P_i = f_i / \sum_i f_i$, and N is the total number of pixels being reconstructed. The observed data $d_1,\ldots,d_m$ is given by the convolutional sum

$$d_j = \sum_{i=1}^{N} A_{ji} f_i + e_j, \quad j = 1,2,\ldots M \qquad (3.2)$$

where $e_j$'s are the additive noise terms with distribution given by $N(0,\sigma_j)$, and $A_{ji}$ represents the point spread function of the imaging system. $A_{ji}$ is a matrix of size MxN and is the convolution matrix of the imaging eq. (3.2).

Let us define the following quadratic form which is the sum of weighted residuals

$$Q(f_1,\ldots f_N) = \frac{1}{2} \sum_{j=1}^{M} \frac{1}{\sigma_j^2} \left( \sum_{i=1}^{N} A_{ji} f_i - d_j \right)^2 \qquad (3.3)$$

The equation (3.3) can be expressed in the vector notation as follows

$$Q(\underline{f}) = \frac{1}{2} (\underline{Af} - \underline{d})^T \operatorname{diag} \left( \frac{1}{\sigma_1^2}, \ldots, \frac{1}{\sigma_m^2} \right) (\underline{Af} - \underline{d}) \qquad (3.4)$$

where,

$$\underline{A} = [A_{ji}]_{MxN}$$

$$\underline{f} = [f_1,\ldots,f_N]^T$$

$$\underline{d} = [d_1, \ldots, d_m)^T$$

and $$D = \text{diag}\left[\frac{1}{\sigma_1^2}, \ldots, \frac{1}{\sigma_1^2}\right].$$

For large values of M, the equation (3.3) can be replaced with the single constraint

$$Q(f_1, \ldots, f_N) = \frac{1}{2} M \tag{3.5}$$

by Law of Large Numbers. The condition $Q = \frac{1}{2} M$ determines the set of feasible images $\underline{f}$, which passes the given statistical test for consistency with the measured data (3.2).

Among the set of feasible images, the maximum entropy criterion selects the one which has the least configurational information.

Thus, we maximize the entropy $H(P_1, \ldots P_N)$ subject to the constraints $Q(f_1, \ldots, f_N) = \frac{1}{2} M$ and $\Sigma f_i = T$, a constant. The second constraint is introduced since the total intensity has a status different from individual pixel values and ensure the law of conservation of energy. Rewriting (3.1) we get

$$H(P_1, \ldots, P_N) = - \sum_{i=1}^{N} \frac{f_i}{\Sigma f_i} \log \frac{f_i}{\Sigma f_i}$$

$$= - \sum_{i=1}^{N} \frac{f_i}{\Sigma f_i} (\log f_i - \log \Sigma f_i)$$

$$= -\sum_{i=1}^{N} \frac{f_i}{\Sigma f_i} \log f_i + \log \Sigma f_i \sum_{i=1}^{N} \frac{f_i}{\Sigma f_i}$$

$$= H(f_1,\ldots,f_N)/\Sigma f_i + \log \Sigma f_i$$

From above , since $\Sigma f_i$ is a constant, it can be proved easily that the following three formulations of the entropy measure are equivalent to each other.

a) Maximize $H(P_1,\ldots,P_N)$ w.r.t. $(f_1,\ldots,f_N)$, subject to constraints $Q = \frac{1}{2} M$ and $\Sigma f_i = T$ (3.6a)

b) Maximize $H(f_1,..,f_N) = -\Sigma f_i \log f_i$ w.r.t. $f_i$, i=1...,N, under the constraints $Q = \frac{1}{2} M$ (3.6b)

and $\Sigma f_i = T$

c) Maximize $[H(f_1,\ldots f_N)+\mu \sum_i f_i - \lambda Q(f_1,\ldots,f_N)]$ subject to

$Q = \frac{1}{2} M$ and $\sum_i f_i = T$ (3.6c)

where $\mu$ and $\lambda$ are the Lagrange multipliers.

We shall solve for case (3.6c) but without the constraint $\Sigma f_i = T$ i.e.

$$\underset{f}{\text{Max.}} \; [H(f_1,\ldots f_N)+\mu\Sigma_i f_i - \lambda Q(f_1,\ldots,f_N] \text{ subject to } Q = \tfrac{1}{2} M. \quad (3.7)$$

The parameter $\mu$ can always be adjusted such that the solution of (3.7) satisfies the constraint of total intensity.

If $f_1^*, \ldots, f_N^*$ is a solution of (3.7), then it also a solution of problems (3.6a), (3.6b) and (3.6c) when $\sum_i f_i^* = T$.

This algorithm for maximization of the entropy measure (3.7) is developed by first considering the unconstrained maximization problem, related to (3.7), as follows

$$\underset{\underline{f}}{\text{Maximize}} \quad [H(f_1,\ldots f_N) + \mu \Sigma f_i - Q(f_1,\ldots,f_N)]$$

$$(= J(\underline{f};\ \mu,\lambda)) \qquad (3.8)$$

Differentiating (3.8) with respect to $\underline{f}$ gives:-

$$\nabla J = \frac{dJ}{d\underline{f}} = \frac{d}{d\underline{f}} \left[ - \sum_{i=1}^{N} f_i \log f_i + \mu \Sigma_i f_i - \lambda Q(f_1,\ldots,f_N) \right]$$

$$= - [\log f_1,\ldots,\log f_N]^T + (\mu-1)[1,\ldots,1]^T - \lambda \frac{dQ(f)}{d\underline{f}} \qquad (3.9)$$

From (3.4)

$$Q = \tfrac{1}{2} (\underline{Af} - \underline{d})^T \underline{D} (\underline{Af} - \underline{d})$$

$$= \tfrac{1}{2} [\underline{f}^T \underline{A}^T \underline{D}\ \underline{A}\ \underline{f} - \underline{d}^T \underline{D}\ \underline{A}\ \underline{f} - \underline{f}^T \underline{A}^T \underline{D}\ \underline{d} + \underline{d}^T\ \underline{D}\ \underline{d}]$$

$$\frac{dQ}{df} = \frac{1}{2}\,[2\,\underline{A}^T\,\underline{D}\,\underline{A}\,\underline{f} - \underline{d}^T\underline{D}\,\underline{A} - \underline{A}^T\,\underline{D}\,\underline{d}\,]$$

$$= \frac{1}{2} * [2\,\underline{A}^T\underline{D}(\underline{A}\underline{f} - \underline{d}\,)] \qquad \left\{\because \underline{A}^T\underline{D}\,\underline{d} = \underline{d}^T\,\underline{D}\underline{A}\right\}$$

$$= \underline{A}^T\underline{D}\,(\underline{A}\underline{f} - \underline{d})$$

Substituting this in (3.9a) we get

$$J = -[\log f_1,\ldots,\log f_N] + (\mu-1)[1,\ldots,1]^T - \lambda\,\underline{A}^T\underline{D}(\underline{A}\underline{f}-d).$$

Equating $\nabla J = 0$ yields the solution of (3.8)

Thus, if $f_1^o,\ldots f_N^o > 0$ is a solution of (3.8) then

$$\nabla J = -[\log f_1^o,\ldots,\log f_N^o\,] + (\mu-1)[1,\ldots,1]^T$$

$$\lambda\,\underline{A}^T\underline{D}\,(\underline{A}\underline{f}^o - \underline{d}) = 0 \qquad (3.9)$$

When $\lambda \geq 0$, the Jacobian of J

$$\nabla^2 J = -\text{diag}\,[\frac{1}{f_o},\ldots,\frac{1}{f_N}] - \lambda\underline{A}^T\underline{D}\,\underline{A} < 0 \qquad (3.10)$$

is negative definite. Hence J is a strictly a concave function.

Lemma 3.1.

Assuming $\lambda \geq 0$, the solution of (3.9) could have at most one solution, and the solution (if it exists) must be the maximal point of the function $J(f;\,\mu,\lambda)$.

Proof: For a strictly concave function, there exists at most

one maximal point. Since $\nabla^2 J < 0$, hence $\nabla J = 0$ is the sufficient condition for a maximal point. Also, it is clear that when $\lambda \geq 0$, problem (3.8) is equivalent to solving the stationary point equation $\nabla J = 0$.

Theorem 3.1

Assume that $\lambda_o \geq 0$ and $\underline{f}^o$ is the solution of $\nabla J(\underline{f};\mu, \lambda_o) = 0$. Then $\mu$, $\lambda_o$, $\underline{f}^o$ uniquelly determine a branch $\underline{f}(\lambda\,;\, \mu, \lambda_o, f^o)$ of maximal points of the function $J(\underline{f},\mu,\lambda)$ corresponding to various values of $\lambda > 0$. This branch of maximal points is governed by a initial value problem of differential equation, given by

$$\nabla^2 J(\underline{f};\mu,\lambda)\ \frac{df}{d\lambda} = \nabla Q(\underline{f})$$

$$\underline{f}(\lambda_o, \mu, \lambda_o, \underline{f}^o) = \underline{f}^o \qquad (3.11)$$

In expanded form, it becomes

$$\left(\text{diag}\left[\frac{1}{f_1}, \ldots, \frac{1}{f_N}\right] + \underline{A}^T\ \underline{D}\ \underline{A}\right) \frac{df}{d\lambda} = -\underline{A}^T\underline{D}\,(\underline{A}\underline{f} - \underline{d})$$

$$\underline{f}(\lambda_o; \mu, \lambda_o, \underline{f}^o) = \underline{f}^o \qquad (3.12)$$

Proof: For $\lambda_o = o$, if $\underline{f}^o$ is the solution then

$$J(\underline{f}^o;\ \mu, \lambda_o) = 0 \text{ and } \nabla^2 J(f^o;\ \mu, \lambda_o) < 0$$

This implies that the initial value problem (3.11) has unique solution curve $\underline{f}(\lambda\ ,\mu,\ \lambda_o,\ f^o)$ where $\lambda_o, \lambda$ belong to an open

interval $[\lambda_1, \lambda_2)$ where $\lambda_1 \geq 0$ and $\lambda_2 \leq \infty$. Along this solution curve the gradient of $\nabla J$ with respect to $\lambda$, $\frac{d}{d\lambda} \nabla J = 0$. Since $\nabla J$ is a function of $\lambda$ and f, hence

$$\frac{d}{d\lambda} \nabla J = \frac{d}{d\lambda} \nabla J(\lambda, f) = \frac{d \nabla J}{df} \cdot \frac{df}{d\lambda} + \frac{d \nabla J}{d\lambda}$$

$$= \nabla^2 J \quad \frac{df}{d\lambda} + \frac{d}{d\lambda} \nabla J$$

Substituting the value of $\nabla J$ in the above equation gives

$$\frac{d}{d\lambda} \nabla J = (\nabla^2 J) \frac{df}{d\lambda} + \frac{d}{d\lambda} [-(\log f_1, \ldots, \log f_N)^T +$$

$$(\mu - 1) [1, \ldots 1]^T - \lambda \underline{A}^T \underline{D} (\underline{Af} - \underline{d})]$$

$$= (\nabla^2 J) \frac{df}{d\lambda} - \underline{A}^T \underline{D} (\underline{Af} - \underline{d})$$

or,

$$\frac{d}{d\lambda} \nabla J = \nabla^2 J \frac{df}{d\lambda} - \nabla Q(f) = 0$$

Integrating from $\lambda_o$ to $\lambda$, we get

$$\nabla J(f(\lambda ; \mu, \lambda_o, f^o); \mu, \lambda) - \nabla J(f(\lambda_o, \mu, \lambda_o, f^o);$$

$$\mu, \lambda_o) = 0$$

i.e. $\nabla J(f(\lambda ; \mu, \lambda_o, f^o); \mu, \lambda) = \nabla J(f^o; \mu, \lambda_o)$

since $\underline{f}(\lambda_o; \mu, \lambda_o, \underline{f}^o) = \underline{f}^o$

But, $\underline{f}^o$ being the solution of (3.8) for $\lambda = \lambda_o$, $\nabla J(f^o; \mu, \lambda_o) = 0$

$\therefore \nabla J(f(\lambda ; \mu, \lambda_o, f^o); \mu, \lambda) = 0$

As a result, $\underline{f}(\lambda\ ;\ \mu,\ \lambda_o,\ f^o)$ is a branch of solution of the stationary point problem (3.8) or a branch of maximal points of the function $J(f;\ \mu,\lambda\ )$ by Lemma 3.1.

Lemma 3.2:

Assume that a branch $\underline{f}(\lambda\ ;\ \mu,\ \lambda_o, f^o)$ is defined by (3.12) where $\nabla J(f^o;\ \mu,\ \lambda_o) = 0$ as before. If there exists a point $\underline{f}^* = \underline{f}(\ \lambda^*;\ \mu,\ \lambda_o, f^o)$ along the branch such that

$$Q(\underline{f}^*) = 0 \tag{3.13}$$

then the whole branch shrinks to a single point $\underline{f}^o$ and for each i: $f_i^o = \exp(\mu-1)$.

Proof:

Along the branch $f(\lambda\ ;\ \mu,\ \lambda_o, f^o)$, we have

$$\nabla J(f(\lambda\ ;\ \mu,\ \lambda_o, f^o),\ \mu,\lambda\ ) = 0$$

or

$$-[\log f_1,\ldots,\log f_N\ ]^T + (\mu-1)[1,\ldots,1]^T = \lambda\nabla Q(\underline{f})$$

From this it is clear that

$Q(\underline{f}^*) = 0$ implies $f_1^* = \ldots = f_N^* = \exp(\mu-1)$ and (3.11) is equivalent to

$$\nabla^2 J(f;\mu,\ \lambda\ )\ \frac{df}{d\lambda} = \nabla Q(f)$$

$$\underline{f}\Big|_{\lambda=\lambda^*} = \underline{f}^*$$

The initial value problem thus has a unique constant solution given by $\underline{f} = \underline{f}^*$.

Theorem 3.2:

Assume that $\lambda_o \geq 0$ and $\underline{f}^o$ is the solution of $J(f; \mu, \lambda_o) = 0$. If for each S,

$$A_S = \begin{bmatrix} A_{1S} \\ \vdots \\ A_{MS} \end{bmatrix} \neq \begin{bmatrix} 0 \\ \vdots \\ 0 \end{bmatrix} \quad ; \; S = 1,\ldots N \qquad (3.14)$$

then the branch $f(\lambda ; \mu, \lambda_o, f^o)$ could be extended to $[o,\infty)$.

Proof:

Let the branch be defined on $[\lambda_1, \lambda_2)$ where $\lambda_1 \geq 0$ and $\lambda_2 \leq \infty$. Along the branch we have

$$\nabla J = 0$$

i.e.

$$-[\log f_1,\ldots,\log f_N]^T+(\mu-1)[1,\ldots,1]^T = \lambda \underline{A}^T\underline{D}\,\underline{A}\,\underline{f} - \lambda \underline{A}^T\underline{D}\,\underline{d} \qquad (3.14a)$$

Let $\underline{P}$ be $\underline{A}^T\underline{D}\,\underline{A}$. From assumption (3.14), it is easy to see that

$$P_{SS} = \sum_{K=1}^{M} \frac{A_{KS}^2}{\sigma_K^2} \neq 0 \; ; \quad S = 1,\ldots,N$$

We could prove that $\underline{f}(\lambda ; \mu, \lambda_o, f^o)$ is bounded from above when $\lambda \to \lambda_2$. Assume the contrary, i.e. if there is a

sequence $\lambda_k \to \lambda_2$ such that

$$f_{S_K}(\lambda_k; \mu, \lambda_o, f^o) = \max_i f_i(\lambda_k; \mu, \lambda_o, f^o) \to \infty \quad \text{as } K \to \infty$$

then from (3.14a), by dividing each term by f, we have

$$-\frac{\log f_{S_K}}{f_{S_K}} + \frac{(\mu-1)}{f_{S_K}} = \lambda_K \sum_{i \neq S_K} P_{S_K i} \frac{f_i}{f_{S_K}} + \lambda_K P_{S_K S_K} - \lambda_K \sum_j A_{j S_K} \frac{d_j}{\sigma_j^2 f_{S_K}}$$

The left hand side tends to zero and the right hand side would be greater than some positive number because $P_{SS} > 0$ and $\lambda_2 > 0$ when $\lambda_K$ is closed to $\lambda_2$. This being a condradiction, we conclude that there exists a positive number $L > 0$ such that for all $\lambda \in [\lambda_1, \lambda_2)$,

$$f_i(\lambda; \mu, \lambda_o, f^o) \leq L \; ; \; i = 1, \ldots N \qquad (3.14b)$$

If $\lambda_2 < \infty$, then (3.14b) would guarantee that the left hand side of (3.14a) is bounded since the right hand side is bounded when $\lambda \in [\lambda_1, \lambda_2)$. In this case, there also exists a positive number $\varepsilon > 0$ such that $\forall \lambda \in [\lambda_1, \lambda_2)$,

$$f_i(\lambda; \mu, \lambda_o, f^o) \geq \varepsilon \; . \qquad (3.14c)$$

Because of (3.14b) and (3.14c) for any sequence $\lambda_K$ there exists a subsequence $\lambda_{K_i}$ and $f_1^*, \ldots f_N^*$ such that

$$\forall_i,\ f_i(\lambda_k;\ \mu,\ \lambda_o,\ f^o) \longrightarrow f_i^*\ (\ell \to \infty\ )$$

Therefore

$\nabla J(f_1^*,\ldots,f_N^*;\ \mu,\ \lambda_2) = 0$ and $\underline{f}^*$ is the unique maximal point of $J(f;\mu,\lambda_2)$ by Lemma 3.1. Since any convergent subsequence $\underline{f}(\lambda_{k_\ell},\ \mu,\ \lambda_o,\ f^o)$ of $\underline{f}(\lambda_k;\ \mu,\lambda_o,\ f^o)$ tends to the same limit point $\underline{f}^*$, the sequence $f(\lambda_k;\ \mu,\lambda_o,\ f^o)$ itself would also tend to $\underline{f}^*$. Therefore

$$\lim_{\lambda\to\lambda_2} \underline{f}(\lambda;\ \mu,\lambda_o,f^o) = \underline{f}^*$$

Thus $f(\lambda;\ \mu,\lambda_o,\ f^o)$ can be extended beyond $\lambda_2$. This leads to the final limit $\lambda_2 = \infty$. Similar arguments leads to $\lambda_1 = 0$.

Cor:

Under the assumptions of theorem 3.2, there exists a positive constant L such that $\forall\lambda \in [o,\infty)$,

$$f_i(\lambda;\ \mu,\lambda_o,\ f^o) \leq L;\quad i=1,\ldots N$$

Remark:

If for some S

$$A_S = \begin{bmatrix} 0 \\ \vdots \\ 0 \end{bmatrix} \qquad (3..5)$$

then, $\nabla J(f_1,\ldots f_N;\ \mu,\lambda) = 0$ if and only if

$\forall_i \neq S$,

$$- \log f_i + (\mu-1) = \sum_{j \neq s} P_{ij} f_j - \sum_j A_{ij} \frac{d_j}{\sigma_j^2}$$

Now $f_s$ is determined from $f_1,\ldots f_{s-1}$, $f_{s+1},\ldots f_N$. Corresponding to (3.15), $A' = [A_1,\ldots,A_{s-1}, A_{s+1},\ldots,A_N]$. Thus, the condition (3.14) is not a troublesome restriction

3.3 SOLUTION OF DIFFERENTIAL EQUATIONS:

When $\lambda_o = 0$, the solution of the stationary point eqn. (3.12) is

$$f_1^o = f_2^o = \ldots = f_N^o = \exp(\mu-1) \qquad (3.16)$$

According to lemma 3.2, the following initial value problem

$$\nabla^2 J(f; \mu, \lambda) \frac{df}{d\lambda} = \nabla Q(\underline{f}) \qquad (3.17)$$

$$\underline{f}\Big|_{\lambda = o} = [\exp(\mu-1),\ldots,\exp(\mu-1)]^T$$

determines a branch $f(\lambda\,;\, \mu, \lambda o, f^o)$ of maximal points of $J(f;\mu,\lambda)$ for $0 < \lambda < \infty$ when $\forall\, s$, $A_s \neq 0$

If it happens that $Q(\exp(\mu-1),\ldots,\exp(\mu-1)) = \frac{1}{2} M$, then $f_1^o = f_2^o = \ldots = f_N^o = \exp(\mu-1)$ is exactly the solution of problem (3.6c), and it's not necessary to solve the initial value problem (3.17). If it is not so, then we have to choose $\mu$ in such a way so that a point $f_1^*,\ldots,f_N^*$ on the branch $f(\lambda\,;\, \mu, o, f^o)$ would satisfy the condition $Q(\underline{f}^*) = \frac{1}{2} M$.

The solution curve of the initial value problem (3.17), corresponding to various values of $\mu$ covers the maximal point set of $J(f_1,\ldots,f_N;\ \mu, \lambda)$ with respect to various $\mu$ and $\lambda$ ($\lambda \geq o$). Moreover, they coincide with each other. Our problem is to choose $\mu$ in order that the corresponding branch $f(\lambda\ ;\ \mu,\ o,\ [\exp(\mu-1),\ldots\exp(\mu-1)]^T)$ will now meet the condition

$$Q(f_1,\ldots,f_N) = \frac{1}{2} M.$$

Theorem 3.3:

Assuming that (3.14) holds, then for $o \leq \lambda < \infty$, either $f_i(\lambda\ ;\ \mu,\ o,\ [\exp(\mu-1),\ldots,\exp(\mu-1)]^T) = \exp(\mu-1);\ i=1,\ldots N$ or

$$\frac{d}{d\lambda}\ Q(f(\lambda\ ;\ \mu,o,[\exp(\mu-1),..,\exp(\mu-1)]^T)) < o \qquad (3.18)$$

Proof:

The branch $f(\lambda\ ;\ \mu,\ o,f^o)$ given by initial value problem (3.17) is defined on $[o,\infty)$ by theorem 3.2. Along the branch we derive

$$\frac{dQ}{d\lambda} = \frac{dQ}{df} \cdot \frac{df}{d\lambda}$$

$$= (\nabla Q)^T \frac{df}{d\lambda}$$

Substituting the value of $\frac{df}{d\lambda}$ from (3.11), we get

$$\frac{dQ}{d\lambda} = (\nabla Q)^T (\nabla^2 J)^{-1} (\nabla Q)$$

Now the following two cases arise:

a) For some $o \leq \lambda^* < \infty$, $\nabla Q(f(\lambda^*; \mu, o, f^o)) = 0$.

In this case, for $o \leq \lambda < \infty$,

$$f_i(\lambda; \mu, o, f^o) = \exp(\mu-1) \text{ by lemma 3.2,}$$

b) $\nabla Q \neq o$ along the branch. In this case, $\frac{dQ}{d\lambda} < o$ since $\nabla^2 J < o$. Hence the proof,

Let us define sets U,V,W as follows:

$$U = \left\{\mu;\ Q(\exp(\mu-1),..,\exp(\mu-1)) = \tfrac{1}{2} M\right\}$$
$$V = \left\{\mu;\ Q(\exp(\mu-1),...,\exp(\mu-1)) > \tfrac{1}{2} M\right\}$$
$$W = \left\{\mu;\quad Q(\exp(\mu-1),...,\exp(\mu-1)) \neq 0\right\} \qquad (3.19)$$

It is amply clear from theorem 3.3 that only those initial value problems would produce meaningful result whose $\mu$ belongs either to set U or to set $(V \cap W)$. In other words, only those values of $\mu$ are useful which results in $\underline{f}(\lambda; \mu, o, f^o)$ meeting the condition $Q(\underline{f}) = \frac{1}{2}M$.

If $\mu \in U$, then $f_1 = f_2 = \ldots f_N = \exp(\mu-1)$ is the solution to problem (3.7) with $\lambda = o$. It is also the solution of problem (3.6a) and (3.6b) with $\sum_{i=1}^{N} \exp(\mu-1) = T$.

If $\mu \in \{V \cap W\}$, then $Q(f(o;\mu,o,[\exp(\mu-1),...,\exp(\mu-1)]^T) > \frac{1}{2} M$ and $Q(f(\lambda;\mu,o,[\exp(\mu-1),...,\exp(\mu-1)]^T)$ decreases

strictly monotonously as $\lambda$ increases from 0. Somewhere $\underline{f}$ would reach a value such that $Q(\underline{f}) = \frac{1}{2} M$. Thus, if for some $\lambda = \lambda^*$, $Q(f(\lambda^*; \mu, o, [\exp(\mu-1), \ldots, \exp(\mu-1)]^T) = \frac{1}{2} M$, then $\underline{f}(\lambda^*, \mu, o, [\exp(\mu-1), \ldots \exp(\mu-1)]^T)$ maximizes the entropy $H(P_1, \ldots, P_N)$ subject to the constraints

$$Q = \frac{1}{2} M \text{ and } \sum_i f_i = \sum_i f_i^* .$$

Computing sets U, V, W

A direct computation leads to a quadratic form

$$Q(\alpha, \ldots, \alpha) = a\alpha^2 + b\alpha + c \qquad (3.20)$$

The quadratic form (3.4) yields

$$a = \frac{1}{2} \sum_{j=1}^{M} \frac{\left(\sum_{i=1}^{N} A_{ji}\right)^2}{\sigma_j^2} \qquad (3.20a)$$

$$b = -\sum_{j=1}^{M} \frac{d_j \sum_{i=1}^{N} A_{ji}}{\sigma_j^2} \qquad (3.20b)$$

$$c = \frac{1}{2} \sum_j \frac{d_j^2}{\sigma_j^2} \qquad (3.20c)$$

It is clear that

$$b^2 \leq 4ac$$

Set U

While computing the set U, the following cases may arise

i) If $a > o$, $b^2-4a(c - \frac{1}{2} M) \geq o$, $c > \frac{1}{2} M$ and $b < o$, then $Q(\alpha,...\alpha) - \frac{1}{2} M = o$ has two positive roots $\alpha_1$ and $\alpha_2$. Thus, the two values of $\mu$ constituting the st U are

$$\exp(\mu_1 - 1) = \alpha_1$$

or $$\mu_1 = 1 + \log \alpha_1$$

and similarly

$$\mu_2 = 1 + \log \alpha_2$$

Hence $U \equiv \{1 + \log \alpha_1, 1 + \log \alpha_2\}$

ii) If $a > o$, $c = \frac{1}{2} M$ and $-\frac{b}{a} > o$, then $Q(\alpha,...,\alpha) - \frac{1}{2} M = o$ has one positive root $\alpha_1$ and one negative root. Since log of negative number is undefined, so we consider only the positive root.

Thus, $U \equiv \{1 + \log \alpha_1\}$

iii) If $a > o$, $c < \frac{1}{2} M$, it has one positive root $\alpha_1$, and $U \equiv \{1 + \log \alpha_1\}$,

iv) If $a = o$ and $c = \frac{1}{2} M$, then U is an infinite open interval i.e.

$$U \equiv \{(-\infty, \infty)\}.$$

Apart from above four cases, $U = \emptyset$, an empty set. Also the fourth case occurs very rarely. In general, U is empty or has two elements at most. This corresponds to no equi-entropy solution or two equi-entropy solutions.

Set V:

$$V = \left\{ \mu;\ Q(\exp(\mu-1),\ldots,\exp(\mu-1)) > \tfrac{1}{2} M \right\}$$

Here also, while computing the set V, the following cases may arise:

i) If $a > o$ and $b^2 - 4a\,(c - \frac{1}{2} M) < o$, then due to the inequality of the quadratic eqn. $Q(\alpha,\ldots,\alpha) - \frac{1}{2} M > o$, V is an infinite open interval

$$V \equiv \left\{ (-\infty, \infty) \right\} \tag{3.20a}$$

ii) If $a > o$ and $b^2 - 4a(c - \frac{1}{2} M) > o$, then $Q(\alpha,\ldots\alpha) - \frac{1}{2} M = o$ has two real roots $\alpha_1$, $\alpha_2$, and due to inequality, the set V will consist of two open intervals.

$$V \equiv \left\{ (-\infty,\ 1+\log(\max(o,\alpha_1))),\ (1+\log(\max(o,\alpha_2)),\infty) \right\} \tag{3.20b}$$

where $Q(\alpha_1,\ldots\alpha_2) = Q(\alpha_2,\ldots,\alpha_2) = \frac{1}{2} M$.

iii) If $a = o$ and $b > o$, then

$$V = \left\{ 1+\log\ (\max(\ o,\ \frac{\frac{1}{2} M - c}{b}\ )),\ \infty \right\}$$

iv) If $a = b = o$, and when $c > \frac{1}{2} M$, then $V = (-\infty,\infty)$ ;

and when $c \leq \frac{1}{2} M$, $V = \emptyset$

v) If $a = o$ and $b < o$, then

$$V = \left\{ (-\infty,\ 1 + \log \max\ (o,\ \frac{\frac{1}{2} M - c}{b}\ )) \right\}$$

The case (iv) occurs rarely. In general, V consists of atmost two open interval.

Set W:

$$W = \left\{ \mu;\ \nabla Q(\exp(\mu-1),\ldots,\exp(\mu-1)) \neq o \right\}$$

Substituting the value of $\nabla Q$, we can write

$$W = \left\{ \mu;\ \exp(\mu-1)\underline{A}^T \underline{D}\ \underline{A} \begin{bmatrix} 1 \\ \vdots \\ 1 \end{bmatrix} - \underline{A}^T \underline{D}\ \underline{d} \neq O \right\}$$

While computing W, the following cases may arise:

i) If rank of $\underline{X} = \left[ \underline{A}^T\ \underline{D}\ \underline{A} \begin{bmatrix} 1 \\ \vdots \\ T \end{bmatrix},\ \underline{A}^T\ \underline{D}\ \underline{d} \right]_{Nx2} = 2,$ (3.20c)

then $W = \left\{ (-\infty,\ \infty) \right\}$

ii) If rank of the matrix defined above = 1 and

$\underline{A}^T\ \underline{D}\ \underline{A} \begin{bmatrix} 1 \\ \vdots \\ 1 \end{bmatrix} \neq O$, then there exists a unique $\alpha$ such that

$$\alpha\ \underline{A}^T\ \underline{D}\ \underline{A} \begin{bmatrix} 1 \\ \vdots \\ 1 \end{bmatrix} = \underline{A}^T \underline{D}\ \underline{d} \qquad (3.20d)$$

**iii)** If rank of X is 1 and also

$$\underline{A}^T \underline{D}\, \underline{A} \begin{bmatrix} 1 \\ \vdots \\ 1 \end{bmatrix} = \underline{0} \text{ , then } W = \emptyset$$

In image restoration problems, the case (iii) occurs very rarely and generally the problem falls under category (i).

3.4 DEVELOPMENT OF THE ALGORITHM:

In practical image restoration problem we never compute the set U as the solution so obtained, though it maximizes the entropy absolutely, is completely devoid of any contrasting features and depicts a uniformly smooth background. Hence, we have to look for a $\mu$ belonging to the set $\{V \cap W\}$.

Rewriting the initial value differential equation (3.12):

$$\left(\text{diag}\left[\frac{1}{f_1}, \ldots, \frac{1}{f_N}\right] + \underline{A}^T \underline{D}\, \underline{A}\right) \frac{df}{d\lambda} = -A^T D\, (\underline{Af} - \underline{d})$$

$$\underline{f}(o; \mu, o, \underline{f}^o) = \underline{f}^o = [\exp(\mu-1), \ldots, \exp(\mu-1)]^T$$

From this differential equation system, we can easily obtain the following iterative equations as follows:

$$(\underline{F} + \underline{A}^T \underline{D}\, \underline{A})\, df = -d\lambda\, \underline{A}^T \underline{D}\, (\underline{Af} - \underline{d})$$

or in the form of an iterative scheme,

$$(\underline{F}^k + \lambda_k \underline{A}^T \underline{D} \underline{A})(\underline{f}^{k+1} - \underline{f}^k)$$

$$= -(\lambda_{k+1} - \lambda_k) \underline{A}^T \underline{D}(\underline{Af} - \underline{d}), \quad k = 0,1,2,\ldots \qquad (3.21)$$

where,

$$\underline{F}^k = \text{diag}\left[\frac{1}{f_1^k}, \ldots, \frac{1}{f_N^k}\right]$$

$$\lambda_o = o \; ; \; \lambda_{k+1} > \lambda_k$$

When each $|\lambda_{k+1} - \lambda_k|$ is small enough, each $\underline{f}^k$ would approximate $\underline{f}(\lambda_k; \mu, o, f^o)$ very well, Rewriting (3.21) as follows

$$(\underline{F}^k + \lambda_k \underline{A}^T \underline{D} \underline{A}) \underline{f}^{k+1} = (2\lambda_k - \lambda_{k+1}) \underline{A}^T \underline{D} \underline{A} \underline{f}^k$$

$$+ [1 \ldots 1]^T + (\lambda_{k+1} - \lambda_k) \underline{A}^T \underline{D} \underline{A} \qquad (3.22)$$

The expression (3.22) represents a large system of linear equations. It can be seen that the structure of the coefficient matrix $(\underline{F}^k + \lambda_k \underline{A}^T \underline{D} \underline{A})$ is such that it is positive definite.

The above system of linear equations can be solved by various iterative schemes, but due to the structure of the coefficient matrix, Gauss- Seidal iterative scheme can be employed to solve (3.22) efficiently. The Gauss-Seidal method [21] is a basic scheme for solving a system of partial differential equations iteratively.

Let $\underline{P}$ represent the coefficient matrix $(\underline{F}^k + \lambda_k \underline{A}^T \underline{D} \underline{A})$. Let $\underline{b}$ represent the vector:

$$\underline{b} = (2\lambda_k - \lambda_{k+1}) \underline{A}^T \underline{D} \underline{A} \underline{f} + [1,\ldots,1]^T + \\ + (\lambda_{k+1} - \lambda_k) \underline{A}^T \underline{D} \underline{d}$$

Setting up the iterative equation for $\underline{f}^{k+1}$:

$$\underline{f}^o = \exp(\mu-1) \begin{bmatrix} 1 \\ \vdots \\ 1 \end{bmatrix}$$

$$f_i^{k+1} = \frac{1}{P_{ii}} \left[ b_i - \sum_{j<i} P_{ij} f_j^{k+1} - \sum_{j>i} P_{ij} f_j^k \right] \qquad (3.23)$$

for $i = 1,\ldots,N$ ; $k = 0,1,2,\ldots$

Since $|\lambda_{k+1} - \lambda_k|$ is appropriately small, $\underline{f}^{k+1}$ should be near to $\underline{f}^k$. This greatly speeds up the convergence of (3.23).

To summarize the algorithm, a flow diagram is presented as shown in Fig. 3.1

From the flow chart it is clear that, having fixed the value $\mu$, we are only varying the value of $\lambda$ to achieve the desired solution. Hence this algorithm is essentially an one dimensional search problem. Due to this reason, the algorithm is more efficient in terms of computing time as compared to the earlier (N+1) dimensional search algorithm. However, the restorations obtained by this algorithm are somewhat inferior.

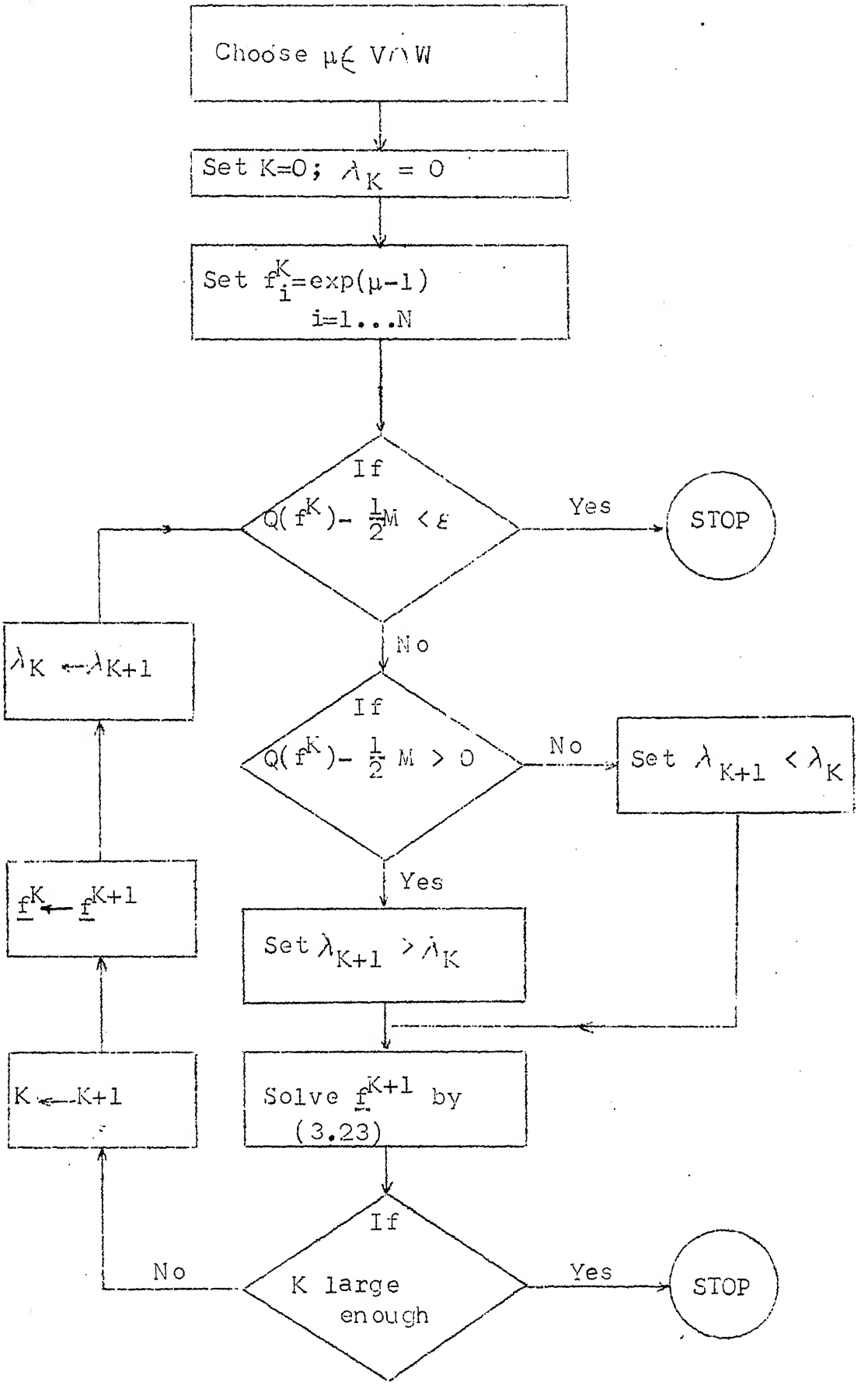



Fig. 3.1: Flow Diagram

Another algorithm which is faster and more efficient than the (N+1) dimensional search algorithm and claims to produce better restorations than the one-dimensional search algorithm ,has been developed by BURCH, GULL and SKILLING[19]. A brief outline of this powerful maximum entropy method is discussed below, though its implementation is beyond the perview of this thesis.

3.5 POWERFUL MAXIMUM ENTROPY METHOD:

This method is applicable to a wide class of problems including that of image restoration when the data are the convolution of an original object with a spatially invariant point-spread function, with the addition of random noise.

The required restoration is a set of positive numbers $f_o, f_1, \ldots f_{N-1}$ which are to be determined, and on which the entropy

$$S = - \sum_{j=o}^{N-1} p(j) \log p(j) \quad p(j) = f(j)/\Sigma f \qquad (3.24)$$

is defined. Related to the set of numbers f(j) is another set of numbers g(i) which represent the simulated data and which would be produced (in the absence of noise) by the observing equipment if the original object were correctly represented by the numbers f. Generally there can be an arbitrary relation between g and f, but the here we will consider the

case of convolution with a known spatially invariant p.s.f. or blurring function h. Thus

$$g(i) = \sum_{j=o}^{N-1} f(j)\, h(i-j) \; ; \; i = 0,1,\ldots N-1$$

where the suffices are written as one dimensional for simplicity. Setting up some statistic to measure the misfit between the actual noisy data d(i) and the simulated data g(i), the most convenient choice is Chi-Squared,

$$X^2 = \sum_{i=1}^{N-1} (g(i) - d(i))^2 / \sigma^2(i) \qquad (3.25)$$

For large N, the plausible values of $X^2$ are all close to $X_o^2 = N$. The condition $X^2 = X_o^2$ now determines the set of feasible images f, which passes the given statistical test for consistency with the actual data. Among the fasible images, maximum entropy criterion selects that particular f which has least configurational information.

The maximum entropy solution can be obtained by maximizing $S(\underline{f})$ over $X^2(f) = X_o^2$. The required maximum of S will be at an extremum of $Q = S - \lambda X^2$ for suitable lagrange multiplier $\lambda$. From this f can be determined via

$$\log A - \log f(j) = (\lambda \sum_j f)\, \partial x^2 / \partial f(j) \qquad (3.26)$$

where

$$A = \exp (\Sigma\, P(i) \log f(i)).$$

The quantity A is a weighted mean of thef(i)and can be treated as the default value to which f(j) tends if there are no data pertaining to cell j ($\partial x^2/\partial f(j) = 0$).

In image restoration, since the total intensity $\Sigma f$ has a status different from individual pixel values, hence $\Sigma f$ forms a second constraint with its own multiplier $\mu$. Thus Q gets modified to $Q = S - \lambda X^2 - \mu \Sigma f$, which in turn modifies A to

$$A = \exp (\Sigma\ p(i) \log f(i) - \mu \Sigma f(i)) \qquad (3.27)$$

One can either choose $\mu$ and hence A, to fit given value of $\Sigma f$ or one can simply use A itself as a predetermined parameter. Generally, the latter choice results in faster implementation of the algorithm. This is equivalent to maximizing a modified form of entropy

$$S = -\Sigma f(j) \log(f(j)/ eA) \qquad (3.28)$$

over $X^2 = X_o^2$, for which the external solution is

$$\log A - \log f(j) = \partial X^2/\partial f(j) \qquad (3.29)$$

The results are not sensitive to the precise value of A.

Normally, such a problem is solved with the aid of a lagrange multiplier $\lambda$ which transforms it to an unconstrained maximization of $Q(f) = S - \lambda X^2$. The simplest such algorithm is steepest ascent, in which one performs a line search along the direction $\nabla Q$. This is however inefficient. The standard way of improving a steepest ascent algorithm is to use some variant of the conjugate gradient technique. Instead of using $\nabla Q$ itself as the search direction, one uses a combination of this with previous gradients choosen to give quick convergence if Q is exactly quadratic in $\underline{f}$. However, Q in maximum entropy is highly non-quadratic and conjugate gradients also proved inadequate. Thus, to make the algorithm more powerful, instead of searching along just one direction at a time, a subspace of three directions was used, namely

$$(e_1)_i = f(i)\,\partial s/\partial f(i)$$

$$(e_2)_i = f(i)\,\partial X^2/\partial f(i)$$

$$(e_3)_i = \sum_j f(i)\frac{\partial^2 X^2}{\partial f(i)\partial f(j)} f(j)\frac{\partial Q}{\partial f(j)}$$

where, in the definition of Q, is given by

$$= (\Sigma f(i)(\partial S/\partial f(i))^2/\Sigma\, f(i)(\partial X^2/\partial f(i))^2)^{1/2}$$

Thus a simultaneous search along three directions is carried out.

Convergence to the image solution is tested by checking that the directions $\nabla X^2$ and $\nabla S$ are parallel to within a

few degrees and that $X^2 = X_o^2$ with less than about 1% error. The number of iterations needed depends on the signal to noise ratio, roughly as its square root, and 15-20 iterations are generally sufficient. Fortunately, this number does not depend on the no. of pixels N being reconstructed so that overall computing time has N log N dependence.

Approximate computing time for images of various sizes are given in Table 1 for various computer systems.

Table 1: ME Computing Time

| | Programmed size | Actual CPU time | Time normalized to 512x512 |
|---|---|---|---|
| IBM 3033 | 256x256 | 8 min | 32 min |
| PDP 11/60 | 512x512 | 20 hr | 20 hr |
| PDP 11/34 | 512x512 | 50 hr | 80 hr |
| PDP 11/34 + AP120B | 512x512 | 2 hr | 2 hr |
| NORD 50/10 | 1024x1024 | 20 hr | 8 hr |
| VAX-11/780 | 512x512 | 3 hr | 5 hr |

# CHAPTER 4

# MINIMUM ENTROPY DECONVOLUTION

## 4.1 INTRODUCTION:

Minimum entropy deconvolution (MED) was developed, by Donoho [22] and Wiggins [23] for the extraction of information about the earth's reflectivity function in the field of Seismology . The input signal in such cases is always consisting of few spikes and the output is a time series which is the convolution of the input signal with the relectivity coefficients of the earth's crest. Thus, the only a-priori information available is the general nature of the input signal. The idea of MED can be adapted for the purposes of digital image restoration for a class of signals which are impulsive and having a non-Gaussian probability density function..The mathematical justification of the principle however remains essentially same.

## 4.2 MINIMUM ENTROPY PRINCIPLE:

Given a time series y, which is a filtered version of a white noise x

$$y = f * x \tag{4.1}$$

where f is the convolution filter, the deconvolution problem is to find a filter b which recovers x from the observed series y, such that

$$x = b * y \qquad (4.2)$$

Since the delay characteristic of the convolution filter f is not generally known, the second-order methods (spectrum, correlation) are unable to solve the deconvolution problem in general. For a typical filter of length p+1, the number of filters with distinct delay characteristics but same second-order property are $2^p$.

Minimum entropy deconvolution is a technique for deconvolution without making prior assumption about the delay characteristic of the filter f. In this approach we seek an operator which, when convolved with the observed signal, converts it into a signal with 'simple' appearance. Here, the term simple implies that the desired signal, recovered from a signal smoothed by convolution with some system function, is impulsive. Such an approach maximizes the order or equivalently minimizes the entropy of the signals. Hence the name Minimum Entropy Deconvolution.

The idea of simple appearance or maximum order or minimum entropy are illustrated in Fig. 4.1. It shows a time series generated according to (4.1), where f is a mixed delay filter. The result y is then filtered using $b^+$, $b^-$, and b filters, all with same second order properties, and correspond to maximum, minimum and mixed delay assumption about f respectively.

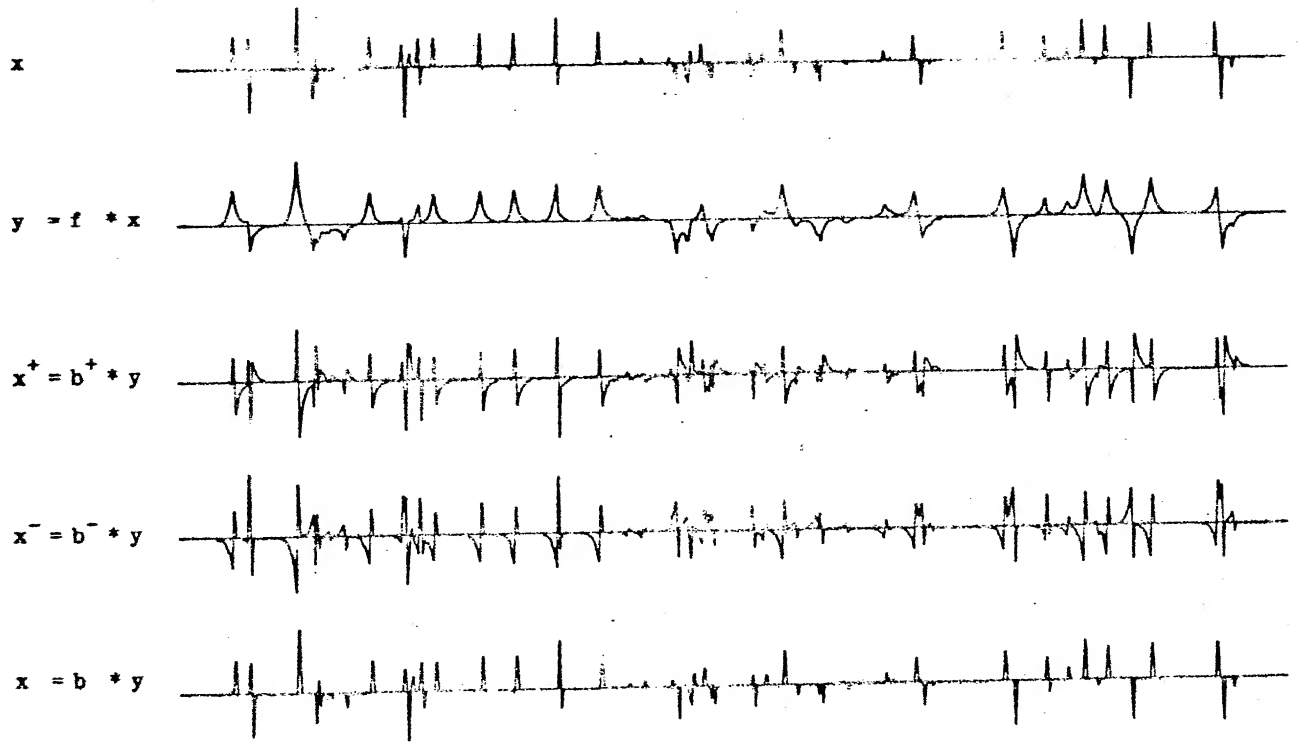



Fig4-1. A time series y = f * x generated using a spiky series for x and a mixed-delay filter for . . Three attempts at deconvolution are shown, corresponding to maximum ($x^+$), minimum ($x^-$), and the correct mixed-delay assumptions on f. The correct deconvolution, x, is also the output with the simplest appearance.

It is apparent that the recovered signal x corresponding to the correct assumption about the delay characteristics of the filter f has the simplest appearance. Thus, the eye, using the judgement of simplicity, can identify the correct solution to the deconvolution problem even though the correlation and the spectrum methods could not. Thus we are looking for a function of the data that reproduces roughly the judgements of simplicity that the eye could make. If O is such an objective function, then our aim is to obtain a formal estimate for the deconvolution problem i.e.

$$\hat{b} = \underset{\tilde{b}}{\text{Maximizer}}\ O(\tilde{b} * y) \tag{4.3}$$

where $\hat{b}$ is the filter which gives $\hat{x} = \hat{b} * y$ the simplest appearance among all series of the form $\tilde{x} = \tilde{b} * y$.

4.3 OBJECTIVE FUNCTIONS:

The objective functions, which work very well for the deconvolution problem discussed above, are

$$\text{i)} \quad O_s^r(\tilde{x}_t) = \frac{\frac{1}{n}\sum_1^n |\tilde{x}_t|^r}{[\frac{1}{n}\sum |\tilde{x}_t|^s]^{r/s}} \tag{4.4}$$

where $\tilde{x}_n$ is the finite sampled version of the estimated series $\tilde{x}$. The objective function

$$O_2^4(\tilde{x}_t) = \frac{\frac{1}{n}\sum_1^n \tilde{x}_t^4}{[\frac{1}{n}\sum_1^n \tilde{x}_t^2]^2} \tag{4.4a}$$

is a special case of (4.4).

ii) The objective function based on an information theoretic point of view is

$$O_E(\tilde{x}_t) = -H\left(\frac{\tilde{x}_t}{s_{\tilde{x}}}\right) \qquad (4.5)$$

where H represents the entropy function and $S_{\tilde{x}}$ is a scaling factor.

All these objective functions have two properties in common.

a) They are scale invariant i.e.

$$O(a\tilde{x}) = O(x) \text{ for } a \neq o.$$

b) They depend on the sample $\hat{x}_n$ only as an unstructured batch of numbers i.e., they depend only upon the empirical distribution of the number $(\tilde{x}(1),\ldots,\tilde{x}(n))$ viewed as a simple random sample.

Due to above properties, the MED type procedure can be successful only if there is a difference between the empirical distribution of $\hat{x}$ and that of $\tilde{x} = \tilde{b} * y$, where $\tilde{b} \neq b$.

Due to the probabilistic structure of the problem, a partial order is introduced which will help in giving fairly complete characterization of the objective functions which

are admissible for use in rule (4.3). For the purposes of analysis, we shall consider the input signal as a time series only. The partial order and the assumptions on the data series which allows this method to work are as follows:

4.4 PARTIAL ORDER $\geq$:

The deconvolved time series $\tilde{x}$ can be written as $\tilde{x} = \tilde{b} * y$. Substituting for y,

$$\tilde{x} = \tilde{b} * f * x$$
$$= (\tilde{b} * f) * x$$

Consequently, each series is sampled from random variables which are linear combinations of independent and identically distributed random variables i.e. $\tilde{x}_t$ is sampled from $\Sigma g_u x_{t-u}$, where $g = \tilde{b}*f$. We shall be discussing the distributional properties of the linear combinations and their implications on model (4.1)-(4.2).

Let $\underline{P}$ be a set of random variables with finite variances which is closed under linear combinations, such that,

$\Sigma a_i P_i + c$ is in $\underline{P}$ for all $P_i \in \underline{P}$

Two random variables P and Q will be regarded as equivalent, written $P \doteq Q$, if for some constants c and $a \neq o$, $aP+c$ has the same probability distribution as Q. $\doteq$ is an equivalence relation on P .

Definition of Parial Order:

$P \dot{\geq} Q$ implies that for appropriate constants $a_i$ with $\Sigma a_i^2 < \infty$,

$$Q \doteq \Sigma a_i P_i$$

where the $P_i$ are independent copies of P. The relation $\dot{>}$ is short for ' $\geq$ but not $\doteq$ '.

$\dot{\geq}$ is a partial order on P because of the following two properties

a) Transitivity:- If $P \dot{\geq} Q$ and $Q \dot{\geq} R$, then

$$P \dot{\geq} R$$

b) Asymmetry:- Let P and Q have finite variances. If $P \dot{\geq} Q$ and $Q \dot{\geq} P$ then $P \doteq Q$.

Proof: Property (a) follows immediately from definition, since if $R \doteq \Sigma b_i Q_i$ and $Q \doteq \Sigma a_i P_i$, then

$$R \doteq \sum_{i,j} a_i b_j P_{ij}$$

To prove property (b), the following characterization is required:

R is a Gaussian random variable if and only if R has finite variance and

$$R \doteq \Sigma a_i R_i \qquad (4.6)$$

holds for some coefficients $\{a_i\}$ which are non trivial, i.e. at least two $a_i$'s $> 0$, and square summable ($\Sigma a_i^2 < \infty$).

Now, suppose $Q \doteq \Sigma\ a_i P_i$ and $P \doteq \Sigma b_j Q_j$. Then $P \doteq \Sigma_{i,j}\ a_i b_j P_{ij}$. This is precisely a relation of the form (4.6). So, if either $\{a_i\}$ or $\{b_j\}$ is nontrivial, then P is Gaussian random variable. Thus, the characterization implies that any linear combination of $P_i$'s, e.g. Q, is also Gaussian, in which case $P \doteq Q$. If the coefficients $\{a_i\}$ and $\{b_j\}$ are both trivial, the $Q \doteq P$ is self evident.

The characterization results provide that if R is Gaussian, then $R \not\cdot P$ for any P, since any linear combination of copies of R is Gaussian, and hence equivalent to R. On the other hand, if P has zero mean and finite variance, then from central limit theorem,

$S_n = n^{-1/2}\ \Sigma P_i$ converges in distribution to a Gaussian random variable R. Since $P \geq S_n$ and $S_n \rightarrow R$ in distribution, we can conclude that $P \cdot\geq R$ for every P in $\underline{P}$. Thus, $\cdot\geq$ is a partial order with this additional stipulation. Hence the proof.

Thus, for P in $\underline{P}$ and R Gaussian,

$$P \cdot\geq \Sigma\ a_i P_i \cdot\geq R \qquad (4.7)$$

(4.7)

This order is strict unless either:

i) P is Gaussian. If so, then $P \doteq \Sigma a_i P_i \doteq R$, or

ii) P is not Gaussian, but the linear combination is trivial (no two $a_i$'s non-zero). In this case $P \doteq \Sigma a_i P_i \to R$.

From relation (4.7), we derive that the linear combinations of independent random variables are more nearly Gaussian than the individual components of the combination. Thus, we introduce a mnemonic 'P $\geq$ Q' implying 'Q is more Gaussian than P'.

The equation (4.7) has a straightforward translation to a time series problem, and we can say that every filtered version of a white noise is more nearly Gaussian than the white noise itself. For example, a white noise p is naturally associated with the random variable P from which $p_t$ is sampled. A filtered version of p, h*p, is associated with $\Sigma h_u p_{t-u}$. Then for filtered white noises p and r, p $\geq$ r is defined to mean P $\geq$ R for associated random variables. Then, (4.7) becomes

$$p \geq h * p \geq r \qquad (4.8)$$

where p is white noise sampled from copies of P in $\underline{P}$, and r is Gaussian time series.

In the deconvolution problem, given by (4.1) and (4.2), the above characterization has immediate application. Since $x = b * y$, and substituting $h = b * f$ in (4.8) we get

$$b * y \geq \tilde{b} * y \geq z \qquad (4.9)$$

where z is a Gaussian time series. Thus, filter b is characterized by giving the least Gaussian output of any filtered version of y. This characterization of b by (4.9) makes the numerical maximization in rule (4.3) possible. It will be seen later that if the objective function O agrees with the partial order $\geq$, then the rule (4.3) is a consistent way of estimating b.

## 4.5 UNIQUENESS OF THE MODEL.

The model, represented by convolutional expressions (4.1) and (4.2), is essentially unique when the representation $y = f * x$ is satisfied only by f and x or by scaled/shifted versions of these.

Lemma 4.1: The model

$$y = f * x \text{ , where} \qquad (4.10)$$

where x is a white noise sampled from x in X and f is an invertible filter, is essentially unique if and only if X is a non-Gaussian random variable.

Proof:

Let there be two representation satisfying (4.10), i.e.

$$y = f * x,$$

$$y = f' * x'$$

Now f * x and f' * x' have the same power spectrum, and both f and f' invertible. Thus we write

$$x' = (f'^{-1} * f) * x$$

$$x = (f^{-1} * f') * x'$$

Both $x \gtrsim x'$ and $x' \gtrsim x$ i.e. $x' \doteq x$.

Now, from relation (4.8) it follows that either X is Gaussian, or $f^{-1} * f'$ is trivial. Hence for the model to be unique, X should be non-Gaussian and the two representation can differ only in scaling and time origin.

To illustrate the non-uniqueness of the convolution model, when x comes from a Gaussian process, see Fig. 4.2. The deconvolutions are affected as in Fig. 4.1. We can see that the three results $x^+$, $x^-$ and x are all uncorrelated and statistically undistinguishable. Also, the eye can not see any qualitative difference between the results. Hence, the judgement of 'simple appearance' is practically impossible if the input signals were Gaussian. It can be now said that eye makes the same sort of qualitative distinctions as dictated by the partial order $\gtrsim$.

## 4.6 CHARACTERIZATION OF ADMISSIBLE OBJECTIVES:

To find out under what conditions, on objective function O, will the estimation rule (4.3) will be consistent,

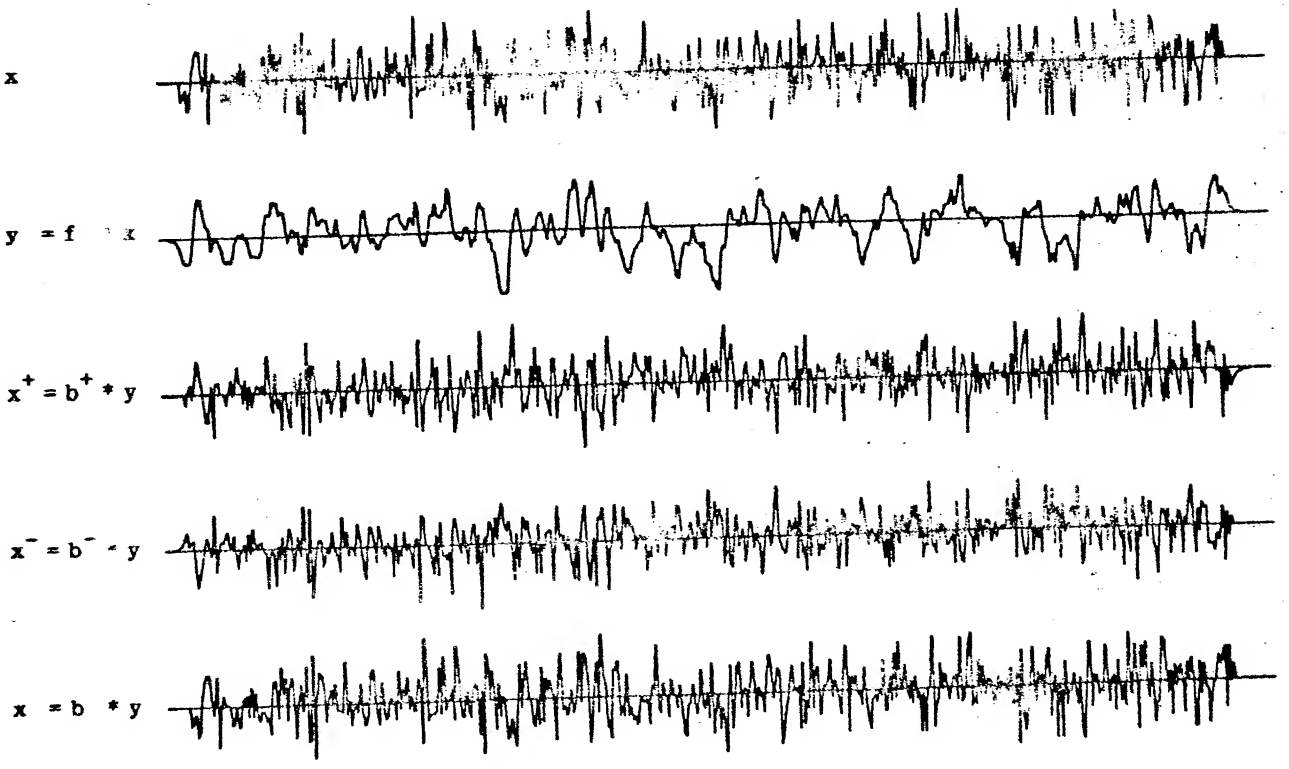



Fig 4.2. The experiment of figure 4.1 repeated with a Gaussian input (x). The visual similarity of the three deconvolutions $x^+$, $x^-$, and x reflects the fact that any Gaussian series can be represented as the result of minimum-, maximum-, or mixed-delay filtering.

it is essential to characterize the admissible objective function. The basic idea is as follows.

By the ergodic theorem, the impirical distribution of $\tilde{X}^n$ converges as n increases to the distribution of the random variable X associated with $\tilde{x}$. Now, $\tilde{x}$ is a good estimate of x to the same extent that $\tilde{X}$ resembles X rather than a linear combination $\Sigma a_i X_i$. So the objective function O has to discriminate between the distribution of X and that of $\Sigma a_i X_i$. Thus, O must do roughly the same thing as the partial order $\rightarrow$.

Definition: Objective function O agrees with $\rightarrow$ in $\mathcal{X}$ if

$$x \rightarrow y \text{ implies } O(x) > O(y) \qquad (4.11)$$

for every x,y which are filtered white noises sampled from random variables in $\mathcal{X}$.

The restrictions on O is that it must be scale invariant since the scale of x and b are unknown and must be continuous function of the distribution of its argument.

Consistency:

The estimation rule (4.3) for a finite length data series can be written as

$$\hat{b}^n = \text{maximizer } O\ (\tilde{b} * y^n) \qquad (4.12)$$

$$\tilde{b} : \text{length } (\tilde{b}) = p+1$$

If the objective function O is scale invariant and a continuous functional, then sequence $(\hat{b}^n)$, defined above, is a consistent sequence of estimates of b for every non-Gaussian y, of the form (4.1), if and only if O agrees with $\rightarrow$ on $\mathcal{X}$.

This shows that an objective which agrees with $\rightarrow$ will give consistent estimates no matter what the distribution of X, as long as it is non-Gaussian.

4.7 IMAGE RESTORATION PROBLEM:

The model described so far can be fitted in many fields related to signal processing. In the field of image processing, the equation (4.1) describes the blurring/degradation of the object image and equation (4.2) describes the deblurring or deconvolution operation for the restoration of the object image.

In this setting, a picture f which has been blurred by an unknown filter h can be recovered, provided the intensity or gray level of the pixels $\underline{f}$ does not follow a Gaussian distribution.

Let $\underline{f}$ represent the digital image vector

$$\underline{f} = f_1, \ldots, f_N$$

and $\underline{h}$ represents the impulse response coefficient vector of the imaging system,

$$\underline{h} = h_1, \ldots, h_L$$

The degraded/blurred image vector $\underline{d}$ is given by

$$d_j = \sum_i f_i \, h_{j-i}, \quad j = 1,\ldots,M \qquad (4.12)$$

where $M = N+L-1$

The vector $\underline{d}$ represents the observation data.

In the absence of the a-priori information about the p.s.f. of the imaging system, the problem of deconvolution or restoration is formidable. But this problem can be tackled when it is known that the object image has a gray level distribution which is non-Gaussian and the gray level variations are impulse like.

In the deconvolution problem we are looking for an inverse filter g of length K which, when convolved with the object image , minimizes the entropy of the estimated data. The order or length of the deconvolution filter has to be small as compared to the length of the observed data because the discrete convolution results in spreading of the data. The resulting deconvolution equation is thus

$$\hat{f}_j = \sum_{i=1}^{M+K-1} d_{j-i} \, g_i \; ; \quad j = 1,\ldots,M+K-1 \qquad (4.13)$$

where K is the order of the deconvolution filter, and $\underline{f}$ is the estimate of object image $\underline{f}$.

4.8 SOLUTION:

The solution of the posed deconvolution problem rests on defining an objective function that measures the simplicity of the desired signal. In other words, we seek a norm which minimizes the entropy such that the order is restored in the estimated data.

The two such objective functions, as discussed earlier and defined by expressions(4.4) and (4.5) are

$$\text{i)}\quad O_V \sum_j \hat{f}_j^4 / \left( \sum_j \hat{f}_j^2 \right)^2 \qquad (4.14)$$

where $\hat{f}_j$ is given by (4.13)

$$\text{ii)}\quad O_E = -[- \sum_j \hat{f}_j \log \hat{f}_j] = -H(\hat{f}_j)$$

$$= \sum_j \hat{f}_j \log \hat{f}_j \qquad (4.15)$$

where $H(\hat{f}_j)$ is a measure of entropy in the estimate $\underline{f}$.

The objective (i) is called Varimax norm, whereas (ii) is called an entropy measure. Maximization of these objective results in minimization of entropy i.e. maximization of the order in the desired signal. The minimization problem is not an unconstrained problem because the image gray levels are always positive and also the total intensity has a status which is different from the intensity

of individual pixel. Thus, the problem develops into

$$\underset{g}{\text{Maximize}} \; (O_V \text{ or } O_E) \quad \text{subject to the constraints}$$

$$\text{a)} \sum_j \hat{f}_j = \sum_j f_i = \sum_i d_i \qquad (4.16)$$

$$\text{b)} \; \hat{f}_j \geq 0, \quad j = 1,\ldots,M+K-1$$

These constraints do not however pose any restriction on the polarity of the coefficients of the deconvolution filter, i.e. they will contain both positive and negative values. The only restriction on deconvolution filter is that its order K should be small such that

$$K \ll M$$

where M is the length of the observed data. It has been observed that a deconvolution filter of order 3 to 5 work well in this context, for the data of different lengths. The constrained optimization of the objective function was achieved by using a NAG library subroutine E04UAF.

# CHAPTER 5

# SIMULATION AND RESULTS

## 5.1 FRIEDEN'S MAXIMUM ENTROPY RESTORATION:

### a) One-Dimensional Data:

Frieden's maximum entropy method, discussed in Chapter 3, has been simulated on DEC-1090 computer system. The object image data of size N, consisting of 32 discrete quantized gray levels, is degraded by passing it through a diffraction blur filter ($\text{sinc}^2(x)$), of constraint length L. The resultant degraded image is of size M = N+L-1. Gaussian noise with zero mean and standard deviation $\sigma$ is generated and added to the degraded image.

From equations (2.15) and (2.16) we get a system of M+1 non-linear equations, which have been solved by the generalized Newton-Raphson iterative method. In this, we start with $\lambda_m = o$ for m=1,...,M, and $\lambda_{M+1}$ is computed from equation (2.15). Now the values of $\lambda_m$, m=1,2,...,M+1, are changed in a direction which satisfy all the M+1 non-linear equations given by (2.15) and (2.16). From the Newton-Raphson method, we obtain the new set of values of $\{\lambda_m\}$ from previous set of values as follows:

$$\begin{bmatrix} \lambda_1 \\ \lambda_2 \\ \vdots \\ \lambda_{M+1} \end{bmatrix}_{new} = \begin{bmatrix} \lambda_1 \\ \lambda_2 \\ \vdots \\ \lambda_{M+1} \end{bmatrix}_{old} - [DEBAF]^{-1} \underline{F} \qquad (5.1)$$

where $\underline{F}$ is a vector of elements $F_i$, i=1,...,M+1 and given by, from eqn. (2.16) and eqn. (2.15) respectively,

$$F_m = \sum_{n=1}^{N} \hat{O}(x_n) h(y_m, x_n) + V(y_m) - B - i(y_m) \qquad (5.2)$$

$$\text{for } m = 1,2,\ldots,M$$

and

$$F_{M+1} = \sum_{n=1}^{N} \hat{O}(x_n) - I_o \qquad (5.3)$$

The matrix [DEBAF] is a (M+1) x (M+1) Jacobian of $\underline{F}$ with respect to the set $\{\lambda_m\}$, and is given by

$$[DEBAF] = \begin{bmatrix} \frac{\partial F_1}{\partial \lambda_1} & \frac{\partial F_1}{\partial \lambda_2} & \cdots & \frac{\partial F_1}{\partial \lambda_{M+1}} \\ \vdots & & & \vdots \\ \frac{\partial F_{M+1}}{\partial \lambda_1} & \frac{\partial F_{M+1}}{\partial \lambda_2} & \cdots & \frac{\partial F_{M+1}}{\partial \lambda_{M+1}} \end{bmatrix} \quad (M+1) \times (M+1) \qquad (5.4)$$

The iterative scheme of eqn. (5.1) can also be expressed as

$$\underline{g} = -[\text{DEBAF}]^{-1}\,\underline{F}$$

where,

$$\underline{g} = \begin{bmatrix} \lambda_1 \\ \vdots \\ \lambda_{M+1} \end{bmatrix}_{new} - \begin{bmatrix} \lambda_1 \\ \vdots \\ \lambda_{M+1} \end{bmatrix}_{old},$$

The above equation can also be modelled as

$$[\text{DEBAF}]\,\underline{g} = -\underline{F} \tag{5.5}$$

This expression represents a system of (M+1) linear equations in (M+1) variables and can be easily solved with the help of NAG Library routines. In this case, F04ARF has been used for solving (5.5). From the solution of (5.5) we can find the new set of values for $\{\lambda_m\}$ by

$$\begin{bmatrix} \lambda_1 \\ \vdots \\ \lambda_{M+1} \end{bmatrix}_{new} = \begin{bmatrix} g_1 \\ \vdots \\ g_{M+1} \end{bmatrix} + \begin{bmatrix} \lambda_1 \\ \vdots \\ \lambda_{M+1} \end{bmatrix}_{old} \tag{5.6}$$

A satisfactory set of values of $\lambda_m$ , m=1,2,...M+1, is obtained after 10 to 40 iterations. However, in this

implementation, about 15 iterations have been found to be sufficient.

Selection of constants $\rho$ and B:

As discussed earlier, (2.8) and (2.9), the constant B should be such that it makes all the noise sample values positive. Since, it is not possible to have a-priori information of each and every noise sample, nor it is possible to find them deterministically, the most convenient value of B is taken as $-2x\sigma$, where $\sigma$ is the deviation of the white Gaussian noise process. It has been found that the value of B contributes largely towards the convergence of the algorithm. If the value of B is chosen larger than required, the maximum value of residual, i.e. $F_i$, after same number of iterations, also becomes large. On the other hand, if B is too small than desired, the process does not converge at all due to negative values of $F_i$. Thus, the optimum value of B has to be found by trial and error. In the present simulation, however, since noise variance is known, the value of B has been kept equal to $2 \times \sigma$ and has been found to be quite suitable.

The relative smoothing of noise and the recovered object image data is controlled by the constant $\rho$ (2.10). For large values of $\rho$ it has been observed that the noise is smoothed out more and hence in the restoration the edges of the

object become sharper. For smaller values of $\rho$, the noise is more abrupt and the edges in the object are smoothened out. Generally, a value of $\rho = 20$ has been found suitable.

Results:

The simulation results for few impulsive objects as well as for randomly stepped objects have been obtained on DEC-1090 computer system. The restorations for various objects for different values of $\sigma$ in $N(0,\sigma)$, and $\rho$ has been obtained and attached as Appendix B. The results so obtained tally with the results published by B.R. Frieden in his paper [1 ]. For restorations obtained by restricting the number of iterations to 15, it takes about 9 second of computer time on DEC-1090 system for N=16 and L=17. For N=24 and L=25, the CPU time requirement is about 20 seconds.

It can be seen from the plots of degraded images and their restorations, that, if the degradation of the object data sequence is considerable, as is the case here, then the smaller peaks get smoothered out, and also if the interval between the two successive pulses is small, then they get superimposed in the restoration. Also, as the size of input data decreases, the restoration becomes increasingly smooth. For this, two objects of same shape but different sizes, i.e. N=16 and 24, were taken. The restoration for N=16 was much

more smooth as compared to N=24. Furthermore, the restorations obtained for impulsive objects have shown better resolution as compared to the randomly stepped objects.

b) Two-Dimensional Data:

The two dimensional object image is represented by a matrix of order N, which is degraded by a two dimensional diffraction blur filter, of constraint size LxL, with seperable impulse response ($\text{sinc}^2(x)\ \text{sinc}^2(y)$). The resultant degraded image of size MxM is further injected with Gaussian noise of zero mean and deviation $\sigma$. For the purpose of processing, the two dimensional data is converted into one-dimensional data by vector-space representation. Thus, from (2.15) and (2.16), where N is replaced by $N^2$ and M is replaced by $M^2$, we get a system of $M^2+1$ non-linear equations, which are solved as before.

Results:

For simulation purposes, an object image of size 8x8 was taken, which represented two bright spots against an uniform background. This was degraded by a filter of constraint size 5x5; and noise $N(0,\sigma)$ was added. The resultant array of size 12x12 was processed to restore the original object image. The value of $\rho$ was chosen to be 20. The convergence was obtained within 10 iteration to the order of

$10^{-4}$. The CPU time required was 3 minutes. The arrays involved for processing this 12x12 blurred image are of size 145x145. This, as well as the iterative nature of the scheme accounts for the excessively large CPU time requirement. For the sake of comparison, the computational time requirement for the processing of a degraded image of size 16x16 is of the order of 40 minutes.

## 5.2 ONE-DIMENSIONAL SEARCH ALGORITHM FOR MAXIMUM ENTROPY RESTORATION:

a) <u>One-Dimensional Data</u>:

The computer implementation of this method involves pre-selecting a value of $\mu$ and then finding a suitable value of $\lambda$ which maximizes the entropy measure (3.8). To achieve meaningful results, $\mu$ must satisfy constraint objective function (3.7). Based on the quadratic form (3.20), the values a,b and c are computed for computing the set V, (3.20a) and (3.20b), and the set W, (3.20c) and (3.20d) . Now, the value of $\mu$ is chosen from $V \cap W$. This initiates the algorithm for the restoration of the degraded image in the presence of noise, as explained in flow chart of Fig. 3.1.

The object image data of length N=16 is passed through a diff raction blur filter of constraint length L. A Gaussian noise with zero mean and deviation $\sigma$ is added.

This degraded image data sequence of length M=32 is processed to restore the original object data.

Results:

The processing for various types of objects indicate that the convergence rate for impulsive objects is very slow and also the restorations so obtained are smoother than desired. However, for randomly stepped objects as well as for objects with gradual variation in intensities, the restoration results are fairly good. Also, the convergence is much quicker for such objects. The computational time requirement is considerably less for this algorithm in comparison to Frieden's algorithm. The increase in the size of the data being processed does not increase the computational time requirement in the same proportions as in the case of Frieden's scheme. However, the resolution achieved is inferior to that of Frieden's algorithm. The results obtained are attached as Appendix B.

b) Two-Dimensional Data:

As before, the two dimensional object data is degraded by passing it through a two-dimensional differaction blur filter and subsequently Gaussian noise is added. For the purpose of processing, the two-dimensional data is converted to one-dimensional form.

Results:

An arbitrary object image data of size 8x8 was degraded by a 9x9 diffraction blur filter . The resultant 16x16 image array was processed to restore the object image data. The computational time requirement was of the order of 1 minute for an object having smooth variation of intensities. However, for impulsive objects the computational time requirement is much more due to the slow convergence.

5.3 MINIMUM ENTROPY DECONVOLUTION:

One-dimensional object data sequences of size N, which are impulsive and hence are intutively known to have non-Gaussian probability distribution, are degraded by diffraction blur filter as before. Our aim is to design a deconvolution filter which, when convolved with the degraded image data restores the object data, minimizes the entropy or maximizes the order in the estimated object data (4.13). The objective function used for the restoration is varimax norm defined by (4.14) i.e.

$$O_v = \sum_j \hat{f}_j^4 \; / \; ( \sum_j \hat{f}_j^2)^2$$

where $\hat{f}_j$ is the estimated object data and is given by

$$\hat{f}_j = \sum_{i=1}^{M+K-1} d_{j-i} \, g_i$$

in which $d_m$ is the degraded image data for $m = 1,\ldots,M$ and $g_k$ is the impulse response coefficients of the deconvolution filter of order K. Substituting $\hat{f}_j$ in the expression for $O_v$, we get

$$O_v = \sum_j \Big( \sum_{i=1}^{M+K-1} d_{j-i}\, g_i \Big)^4 \Big/ \Big( \sum_j \Big( \sum_{i=1}^{M+K-1} d_{j-i}\, g_i \Big)^2 \Big)^2$$

This objective function is maximized under the constraints

$$\sum_{j=1}^{M+K-1} \hat{f}_j = \sum_{i=1}^{M} d_i = \sum_{n=1}^{N} f_n$$

and $\hat{f}_j \geq 0$ ,

for designing the deconvolution filter coefficients $g_k$, $k = 1,\ldots K$. The restored data is then computed from (4.13). The constraint $\hat{f}_j \geq 0$ ensures that the pixel values of the restored object data are positive. The other constraint preserves the amplitudes.

Results:

The simulation results show that for very badly degraded images, the restoration of the original object data is excessively smooth. However, for the images which are degraded to lesser extents, by choosing the degrading filter

of smaller constraint lengths, it is possible to recover the original data by minimum entropy deconvolution. It has been observed from the restoration results that the deconvolution filters of smaller order, 3 to 5, give much better resolution as compared to the deconvolution filters of higher orders. This is understandable because the deconvolution filter tends to increase the spread of the estimated data, and thus the deconvolution filter of smaller order produces better resolution.

Since the solution to the deconvolution problem is obtained iteratively and also the knowledge of point spread function of of the imaging system is completely unknown, the convergence of this scheme is slow. To process a one-dimensional array of length 32, the CPU time requirement on the DEC 1090 system was of the order of 50 seconds. The various results so obtained are attached at Appendix B.

# CHAPTER 6

# CONCLUSIONS

The ideas of maximum entropy method, for the restoration of images, are intrigining since they suggest that one can in principle remove the presence of artifacts, reduce the apparent noise and give extra resolution. Maximum entropy restoration technique is more fundamental than the other techniques and deals with the problem in a very general way. What the maximum entropy method sets out to do is to seek the most likely answer to a particular question that can be obtained from a given body of scientific data: Entropy maximization is a kind of insurance policy that protects the restoration against spurious details for which their is no evidence in the observed data.

Amongst the two maximum entropy approaches considered in Chapter 2 and 3 it has been observed that the Frieden's maximum entropy method gives better results in terms of the resolution achieved as well as the rate of convergence of the algorithm, whereas the one-dimensional search algorithm is computationally faster for larger arrays. The efficiency of the Frieden's algorithm lies in the fact that it seeks the minimization of the error in each pixel of the restored image, whereas the one-dimensional search approach minimizes

the average error in the restored image. Also, the convergence rate for the impulsive data images is exceptionally slow in case of one-dimensional search algorithm based on the solution of initial value differential equations. The simulation results indicate that the one-dimensional search algorithm is not efficient enough to offset the disadvantage of excessive computational time requirement in the case of (M+1) dimensional search algorithm.

The ideas of minimum entropy deconvolution apparently does not offer any promise in the field of image processing. But, its extensive application in the field of seismic data processing does indicate its applicability in the field of image restoration for a class of image signals which are impulsive and have essentially a non-Gaussian probability distribution. The principle of minimum entropy deconvolution, for such a class of image signals, has been applied for the restoration of the object images from diffraction blurred images. In the absence of any a-priori information about the point spread function of the imaging system, the results obtained are quite encouraging. However, if the degradation suffered by the object image is very profound, as well as if the noise level is high, the minimum entropy deconvolution method fails to recover the object image.

# REFERENCES


1. B.R. Frieden, 'Restoring with Maximum Likelihood and Maximum Entropy', J. Opt. Soc. Am. 62, 511 (1974).

2. B.R. Frieden, 'Image Enhancement and Restoration', in T.S. Huang edited Topics in Applied Physics, vol. 6 - Picture Processing and Digital Filtering. 229 (1975).

3. B.R. Frieden, 'Image Restoration by Discrete Convolution of Minimal Length', J. Opt. Soc. Am. 64, 682 (1974).

4. B.R. Frieden, 'Information and the Restorability of Images', J. Opt. Soc. Am. 60, 575 (1970).

5. S.J. Wernecke, 'Maximum Entropy Image Reconstruction', IEEE Transactions on Computers, vol. C.26, No. 4, 351 (1977).

6. R. Kikuchi, B.H. Soffer, 'Max. Entropy Image Restoration I The Entropy Expression' in R. Shaw edited Image Analysis and Evaluation. 226 (1976).

7. C.B. Smith, 'A Dual Method for Maximum Entropy', IEEE Transactions on PAMIPAMSI. 411 (1979).

8. R.K. Bryan, J. Skilling, 'Deconvolution by Maximum Entropy', Mon. Not. R. Astron. Soc., 191, 69 (1980).

9. E.T. Jaynes, 'On the Rationale of Maximum Entropy Methods', IEEE Proceedings, vol. 70, no. 10, Sept. (1982).

10. J.N. Kapur, 'Maximum Entropy Models in Science and Engineering', Part III, 23.18.

11. J.E. Shore, 'Axiomatic Derivation of the Principle of Maximum Entropy and the Principle of Minimum Cross--Entropy', IEEE Trans. on Information Theory, vol. IT-26, No. 1, 26 (1980).

12. B.R. Frieden and W. Swindell, SCIENCE, vol. 191, No. 4233, 26 Mar 1976.

13. S.J. Wernecke, 'Maximum Entropy Techniques for Diagonal Image Reconstruction', in R. Shaw edited Image Analysis and Evaluation, 238 (1976).

14. T. Elfving, 'On some Methods of Entropy Maximization and Matrix Scaling', Linear Algebra and Its Applications. 34, 32, (1980).

15. J. Eriksson, 'A Note on Solution of Large Sparse Maximum Entropy Problems with Linear Equality Constraints', Mathematical Programming 18, 146(1980).

16. B.R. Frieden, 'Statistical Models for the Image Restoration Problem', Computer Graphics and Image Processing 12, 40 (1980).

17. X. Zhuang, K.B. Yu and R.M. Haralick, 'A New Approach to the Solution of the Maximum Entropy Image Reconstruction Problem', Dept. of EE, Virginia Polytechnic Institute of State University, Blacksburg, VA 24061.

18. S.F. Gull and G.J. Daniell, 'Image Reconstruction from Incomplete and Noisy Data', Nature, vol. 272, 686(1978).

19. S.F. Burch, S.F. Gull and J. Skilling, 'Image Restoration by a Powerful Maximum Entropy Method', Computer Vision, Graphics and Image Processing 23, 113(1983).

20. A.J. Levy, 'A Fast Quadratic Programming Algorithm for Positive Signal Restoration', IEEE Trans. on Acoustics, Speech and Signal Processing, 1337(1983).

21. J.N. Franklin, Matrix Theory, Prentice-Hall, Inc., 1968.

22. D. Donoho, 'On Minimum Entropy Deconvolution' in Applied Time Series Analysis II, ' edited by D.F. Findley, Academic Press, 565 (1981).

23. R.A. Wiggins, 'Minimum Entropy Deconvolution', Geoexplaration' 16, 21 (1978).

24. W.K. Pratt, Digital Image Processing, John Wiley and Sons, 1978.

25. A. Rosenfeld and A.C. Kak, O Digital Picture Processing, Second Edition, vol. 1, Academic Press, 1982.

26. D.P. Macadam, 'Digital Image Restoration by Constraint Deconvolution', J. Opt. Soc. Am. 60, 1617(1970).

27. F.B. Hildebrand, Introduction to Numerical Analysis. McGraw Hill Book Co., New York,1950.

# APPENDIX A

## DIGITAL IMAGE CHARACTERISATION

In digital image processing system one usually deals with arrays of numbers obtained by spatially sampling points of a physical image. After processing another array of numbers is produced and these numbers are then used to reconstruct a continuous image for viewing. Image samples normally represent some physical measurements of a continuous image field. For example, measurements of the image intensity or photographic density.

For the purpose of analysis, it is convenient to convert the image matrix to vector form by column or row scanning and then stringing the elements together in a long vector. Consider an image matrix F such that

$$F = \begin{bmatrix} F(1,1) & F(2,1) & \cdots & F(N_2,1) \\ F(2,1) & & & \vdots \\ \vdots & & & \\ F(N_1,1) & & \cdots & F(N_1,N_2) \end{bmatrix}_{N_1 \times N_2} \qquad (A.1)$$

An equivalent scanning operation can be expressed in quantitative form by the use of an $N_2 \times 1$ operational vector $V_n$ and $N_1N_2 \times N_1$ matrix $N_n$ defined as

$$V_n = \begin{bmatrix} 0 \\ \vdots \\ 0 \\ 1 \\ 0 \\ \vdots \\ 0 \end{bmatrix} \begin{matrix} 1 \\ \vdots \\ n-1 \\ n \\ n+1 \\ \vdots \\ N_2 \end{matrix} \qquad N_n = \begin{bmatrix} 0 \\ \vdots \\ 0 \\ I \\ 0 \\ \vdots \\ 0 \end{bmatrix} \begin{matrix} 1 \\ \vdots \\ n-1 \\ n \\ n+1 \\ \vdots \\ N_2 \end{matrix} \qquad (A.2)$$

where each element of $N_n$ represents a $N_1 x N_1$ matrix. Then the vector representation of the data matrix F is given by the stacking operation

$$f = \sum_{n=1}^{N_2} N_n \, F \, V_n \qquad (A.3)$$

In essence, the vector $V_n$ extracts the nth column from F and the matrix $N_n$ places this column into the nth segment of the vector f. Thus f contains the column scanned elements of F. The inverse relation of converting the vector f into matrix form is obtained from

$$F = \sum_{n=1}^{N_2} N_n^T \, f \, V_n^T \qquad (A.4)$$

With the matrix to vector operator of eq. (A.3) and the vector to matrix operator of eq.(A.4), it is easy to convert between vector and matrix representation. The

advantages of dealing with images in vector form are a more compact notation and formulation of a tedious two-dimensional problem into a one-dimensional problem. Thus, the results for one-dimensional signal processing applications can be easily applied to two-dimensional case.

FINITE AREA SUPERPOSITION OPERATOR

Considering the discrete superposition of a spatially truncated data array $F(n_1,n_2)$ for $n_1,n_2 = 1,2,\ldots N$ with a spacially truncated impulse response operator $H(\ _1,\ _2;\ m_1,m_2)$ for $\ _1,\ _2 = 1,2,\ldots L$. In general, the impulse response array changes form for each point $(m_1,m_2)$ in the processed array Q. The finite area superposition operation is defined as

$$Q(m_1,m_2) = \sum_{n_1=1}^{m_1} \sum_{n_2=1}^{m_2} F(n_1,n_2)H(m_1-n_1+1,m_2-n_2+1;\ m_1,m_2) \qquad (A.5)$$

for $m_1,m_2 = 1,2,\ldots M$, where H and F are assumed to be zero outside their range of indices. From the indices of the impulse response array at its external position indicates that $M = N+L-1$, and hence the processed array Q is of larger dimension than the data array F.

In vector-space notation, the finite superposition operation can be written as

$$q = Df \tag{A.7}$$

where D is $M^2xN^2$ matrix containing the elements of the impulse response. It is convenient to partition the superposition operator matrix D into submatrices of dimension MxN, such that

$$D = \begin{bmatrix} D_{1,1} & 0 & & 0 \\ D_{2,1} & D_{2,2} & & \vdots \\ \vdots & \vdots & & \vdots \\ \vdots & \vdots & & \vdots \\ D_{L,1} & D_{L,2} & & D_{M-L+1,N} \\ 0 & D_{L+1,2} & & \vdots \\ 0 & 0 & & \vdots \\ \vdots & \vdots & & \vdots \\ \vdots & \vdots & & \vdots \\ 0 & 0 & & D_{M,N} \end{bmatrix} \tag{A.8}$$

The general nonzero term of D is then given by

$$D_{m2,n2}(m_1,n_1)=H(m_1-n_1+1,m_2-n_2+1;m_1,m_2) \tag{A.9}$$

where $1 \leq n_i < N$ and $n_i \leq m_i \leq n_i+L-1$. Thus, D is highly structured and quite sparse, with the centre band of submatrices containing stripes of zero elements.

If the impulse response is position invariant, then the structure of D does not explicitly depend on the output

array coordinate $(m_1, m_2)$. Also,

$$D_{m2,n2} = D_{m2+1,n2+1} \tag{A.10}$$

Thus, the columns of D are shifted versions of the first column. Under these conditions, D is known as finite area convolution operator.

For a spacially invariant and seperable impulse response

$$H = h_C \; h_R^T \tag{A.11}$$

where $h_R$ and $h_C$ are column vectors representing row and column impulse responses, then

$$D = D_c \times D_R \tag{A.12}$$

The matrices $D_R$ and $D_c$ are MxN matrices of the form

$$D_R = \begin{bmatrix} h_R(1) & 0 & 0 \\ h_R(2) & h_R(1) & 0 \\ \cdot & h_R(2) & h_R(1) \\ \cdot & \cdot & \cdot \\ \cdot & \cdot & \cdot \\ h_R(L) & \cdot & h_R(2) \\ 0 & h_R(L) & \cdot \\ 0 & 0 & \cdot \\ \cdot & \cdot & \cdot \\ 0 & 0 & h_R(L) \end{bmatrix}_{MxN}$$

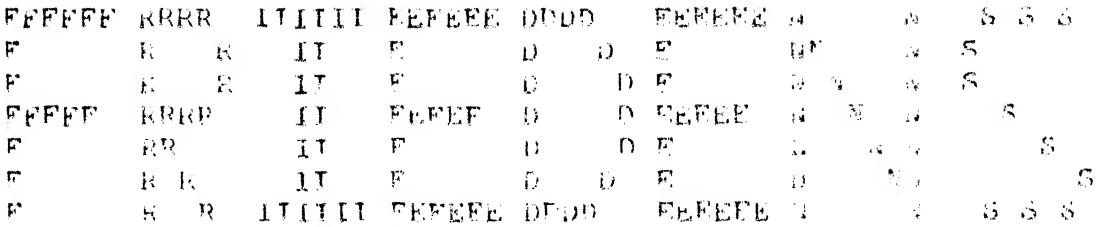
The two-dimensional convolution operation may then be computed by sequential row and column one-dimensional convolutions. Thus

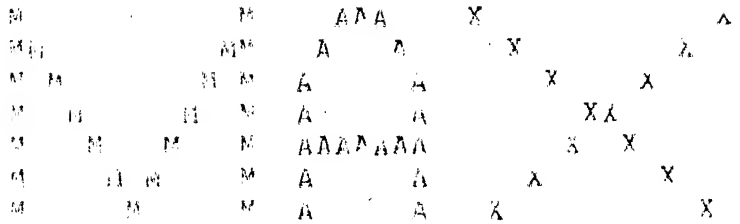
$$Q = D_C F D_R^T \tag{A.13}$$

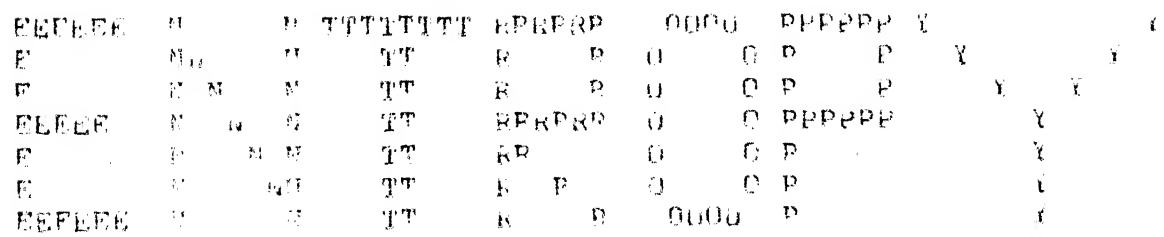
In vector form, finite area superposition or convolution operater requires $N^2L^2$ operations if the zero multiplication of D are avoided. The seperable operator can be computed with only NL(M+N) operations.

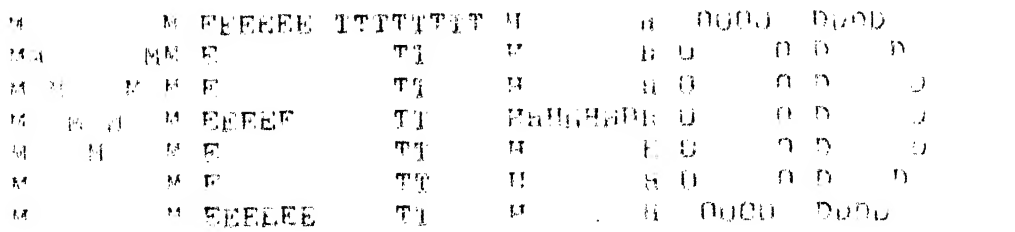
# APPENDIX B

The following pages B-2 to B-123 are divided into three subheads pertaining to three different image restoration techniques under perview. Each subhead constain the FORTRAN-10 listings of the simulation programs followed by the plots of the simulation results. The various programs and the results contain sufficient comments for easy assimilation.









```
C-----------------------------------------------------------
C PROGRAM BLUR.FOR FOR DEGRADING THE OBJECT IMAGE DATA.
C
C THIS PROGRAM DEGRADES THE OBJECT IMAGE DATA BY PASSING
C
C IT THRO' A  DIFFRATION BLUR FILTER(SINC(X)**2).THE FILTER
C
C COEFFS ARE NORMALISED SUCH THAT THEIR SUM IS EQUAL TO 1.
C-----------------------------------------------------------
      PROGRAM BLUR.FOR
      INTEGER PIC(16),IMG(32),NX,LX,MX
      REAL LIML,LIMR,HR(17),DR(32,16),ALPHA,SUM,XIMG(32)
      NX=16
      LX=17
      MX=NX+LX-1
      OPEN (UNIT=21,DEVICE='DSK',FILE='PIC.DAT')
      READ (21,*),(PIC(I),I=1,NX)
      LIML=-3.14159
      LIMR=3.14159
      SUM=0.0
      DO 10 I=1,LX
      ALPHA=LIML+1*(LIMR-LIML)/LX
      HR(I)=(SIN(ALPHA)/ALPHA)**2
      SUM=SUM+HR(I)
   10 CONTINUE
      DO 20 I=1,LX
      HR(I)=HR(I)/SUM
   20 CONTINUE
      DO 40 I=1,MX
      DO 50 J=1,NX
      DR(I,J)=0.0
   50 CONTINUE
   40 CONTINUE
      DO 60 J=1,NX
      DO 70 I=1,LX
      DR(I+J-1,J)=HR(I)
   70 CONTINUE
   60 CONTINUE
      WRITE (45,80),((DR(I,J),J=1,NX),I=1,MX)
   80 FORMAT(16(F10.6,1X))
      DO 100 I=1,MX
      SUM=0.0
      DO 110 K=1,LX
      SUM=SUM+DR(I,K)*PIC(K)
  110 CONTINUE
      IMG(I)=IFIX(SUM)
      XIMG(I)=SUM
  100 CONTINUE
      WRITE(40,120) (IMG(I),I=1,MX)
      WRITE(50,*) (XIMG(I),I=1,MX)
  120 FORMAT(16(I6,1X))
      STOP
```

```
------------------------------------------------------------
THE PROGRAM NOISE.FOR ADDS WHITE GAUSSIAN NOISE N(0,SIGMA)

TO THE BLURRED IMAGE DATA.
------------------------------------------------------------
PROGRAM NOISE.FOR
INTEGER IMG(32),NX,LX,MX,ISUM,PIC(16)
REAL SIGMA,MEAN,SUM,XIMG(32),SNR
NX=16
LX=17
MX=NX+LX-1
OPEN (UNIT=21,DEVICE='DSK',FILE='PIC.DAT')
OPEN (UNIT=22,DEVICE='DSK',FILE='FOR50.DAT')
READ(21,*) (PIC(I),I=1,NX)
READ(22,*) (XIMG(I),I=1,MX)
ISUM=0
DO 10 I=1,NX
ISUM=ISUM+PIC(I)**2
CONTINUE
SIGMA=0.10
MEAN=0.0
ISUM=ISUM/NX
SNR=10*ALOG10(ISUM/SIGMA**2)
DO 20 I=1,MX
SUM=XIMG(I)+GOSDDF(MEAN,SIGMA)
IMG(I)=IFIX(SUM)
XIMG(I)=SUM
CONTINUE
WRITE(41,30) (IMG(I),I=1,MX)
WRITE(51,*) (XIMG(I),I=1,MX)
WRITE(57,77) SNR
FORMAT(10X,'SNR IS =',F10.4)
FORMAT(16(I6,1X))
STOP
END
------------------------------------------------------------
------------------------------------------------------------
```

```
------------------------------------------------------------
PROGRAM ME.FOR FOR FRIEDEN'S MAXIMUM ENTROPY SOLUTION.

THIS PROGRAM SOLVES A SYSTEM OF NON-LINEAR EQUATIONS TO

RESTORE THE ORIGNAL OBJECT IMAGE DATA FROM THE DEGRADED

IMAGE DATA.
------------------------------------------------------------
PROGRAM ME.FOR
REAL B,F(33),DF(33,33),L(33),SUM1,SUM2,SUM3,PO,SIGMA,MAX
,S(32,16),SUM(16),IO,IMG(32)
INTEGER SINT(32,2),NOOFI,MX,NX,LX,M1,M2,M3,P,PIC(16)
COMMON S,SINT,F,DF,L,SUM,MAX,IMG,B,NO,MX,LX,NX,IO
SIGMA=0.2
B=2.0*SIGMA
NX=16
LX=17
MX=NX+LX-1
NOOFI=15
PO=20.0
OPEN(UNIT=21,DEVICE='DSK',FILE='FOR51.DAT')
READ(21,*) (IMG(I),I=1,MX)
OPEN(UNIT=22,DEVICE='DSK',FILE='FOR45.DAT')
READ(22,*)((S(I,J),J=1,NX),I=1,MX)
S IS THE CONVOLUTION MATRIX OF THE IMAGING SYSTEM.
DO 10 I=1,LX
SINT(I,1)=I
SINT(I,2)=1
CONTINUE
SINT(LX,1)=LX-1
DO 20 I=LX+1,MX
SINT(I,1)=MX-I+1
SINT(I,2)=I-LX+1
CONTINUE
IO=0.0
DO 30 I=1,MX
IO=IO+IMG(I)*1.0
CONTINUE
DO 40 I=1,MX
L(I)=0.0
CONTINUE
L(MX+1)=-1-ALOG(IO*1.0/MX)
DO 50 I=1,MX
SUM(I)=EXP(-1-L(MX+1))
CONTINUE
CALL GETF
CALL GETDER
DO 60 I=1,NOOFI
CALL GETLAM
CALL GETSUM
CALL GETF
```

```
DO 140 I=1,MX
PIC(I)=IFIX(SUM(I))
CONTINUE
WRITE(48,150)(PIC(I),I=1,MX)
FORMAT(16(I6,1X))
STOP
END
```

---

SUBROUTINE GETF COMPUTES THE RESIDUAL OF THE CONSTRAINTS

AFTER EACH ITERATION.F(I),I=1..MX+1,SHOULD BE NEARLY ZERO

AFTER THE SPECIFIED NO. OF ITERATIONS.

---

```
SUBROUTINE GETF
REAL SUM1,S,F,DF,D,SUM,MAX,B,RO,TO,IMG
INTEGER SINT,LX,IA,MX,K,KA
COMMON S(32,16),SINT(32,2),F(33),DF(33,33),D(33),SUM(16)
,MAX,IMG(32),B,RO,MX,DX,MA,10
MAX=0.0
DO 10 I=1,MX
F(I)=0.0
J=SINT(1,2)
DO 20 M=1,SINT(1,1)
F(I)=F(I)+S(I,J)*SUM(J)
J=J+1
CONTINUE
F(I)=F(I)+EXP(-1-D(I)/RO)-B-IMG(I)
IF(MAX.GE.F(I)) GOTO 10
MAX=F(I)
CONTINUE
F(MX+1)=0.0
DO 30 I=1,MX
F(MX+1)=F(MX+1)+SUM(I)
CONTINUE
F(MX+1)=F(MX+1)-TO
IF(MAX.GE.F(MX+1)) GOTO 40
MAX=F(MX+1)
CONTINUE
RETURN
END
```

---

SUBROUTINE GETDEB COMPUTES THE JACOBIAN OF VECTOR F WITH

RESPECT TO THE LAMDA'S.

---

```
SUBROUTINE GETDEB
INTEGER SINT,DX,MA,MX
```

```
,MAX,IMG(32),B,RO,NX,LX,MX,10
DO 10 I=1,MX
DO 20 J=1,MX
DF(I,J)=0.0
K=SINT(J,2)
DO 30 M=1,SINT(J,1)
DF(I,J)=DF(I,J)-S(J,K)*S(I,K)*SUM(K)
K=K+1
CONTINUE
IF(I.NE.J) GOTO 20
DF(I,J)=DF(I,J)-(EXP(-1-L(I)/RO))/PO
CONTINUE
CONTINUE
DO 40 I=1,MX
DF(I,MX+1)=0.0
J=SINT(I,2)
DO 50 K=1,SINT(I,1)
DF(I,MX+1)=DF(I,MX+1)-S(I,J)*SUM(J)
J=J+1
CONTINUE
CONTINUE
DO 60 I=1,MX
DF(MX+1,I)=0.0
J=SINT(I,2)
DO 70 K=1,SINT(I,1)
DF(MX+1,I)=DF(MX+1,I)-S(I,J)*SUM(J)
J=J+1
CONTINUE
CONTINUE
DF(MX+1,MX+1)=-F(MX+1)+10
RETURN
END
```

---

THE SUBROUTINE GETSUM COMPUTES THE ESTIMATED OBJECT DATA

AFTER EACH ITERATION.

---

```
SUBROUTINE GETSUM
INTEGER SINT,LX,MA,MX
REAL SUM1,S,F,DF,L,SUM,MAX,B,RO,TO,IMG
COMMON S(32,16),SINT(32,2),F(33),DF(33,33),L(33),SUM(16),
MAX,IMG(32),B,RO,NX,LX,MX,TO
DO 10 I=1,MA
SUM(I)=0.0
J=I
DO 20 K=1,LX
SUM(I)=SUM(I)-L(J)*S(J,I)
J=J+1
CONTINUE
SUM(I)=SUM(I)-1-L(MX+1)
SUM(I)=EXP(SUM(I))
CONTINUE
```

```
------------------------------------------------------------
SUBROUTINE GETLAM SOLVES A SYSTEM OF MX+1 LINEAR EQS.TO

COMPUTE THE NEW SET OF LAMDA'S.
------------------------------------------------------------
SUBROUTINE GETLAM
INTEGER SUM1,LX,MA,MX
REAL SUM1,S,F,DF,L,SUM,MAX,B,RO,DL(33),LO,SUM2,LMO
COMMON S(32,16),SINT(32,2),F(33),DF(33,33),L(33),SUM(16)
,MAA,LMO(32),B,RO,MA,LX,MX,LO
REAL XDF(33,33),XF(33),WKSPCE(33)
INTEGER IA,N,IFAIL
DO 200 I=1,MX+1
DO 210 J=1,MX+1
XDF(I,J)=DF(I,J)
CONTINUE
CONTINUE
DO 220 I=1,MX+1
XF(I)=-1*F(I)
CONTINUE
IA=MX+1
IFAIL=0
N=MX+1
CALL F04ARF(XDF,IA,XF,N,DL,WKSPCE,IFAIL)
IF(IFAIL.NE.0)STOP
DO 230 I=1,MX+1
L(I)=L(I)+DL(I)
CONTINUE
RETURN
END
------------------------------------------------------------
------------------------------------------------------------
```

```
C-----------------------------------------------------------
C PROGAM PLOT.FOR TO PLOT THE IMAGE DATAS.
C-----------------------------------------------------------
      PROGRAM PLOT.FOR
      INTEGER MX,NX,LX,PIC(32)
      REAL WB(64)
      COMMON NX,LX,MX,PIC,WB
      NX=16
      LX=17
      MX=NX+LX-1
      OPEN(UNIT=21,DEVICE='DSK',FILE='PIC.DAT')
      READ(21,*)(PIC(I),I=1,NX)
      PRINT 10
   10 FORMAT(1H ,///////,40X,'THE OBJECT IMAGE')
      CALL PPLOT
      OPEN(UNIT=22,DEVICE='DSK',FILE='FOR40.DAT')
      READ(22,*)(PIC(I),I=1,MX)
      PRINT 20
   20 FORMAT(1H1,///////,30X,'DIFFRACTION BLURRED IMAGE')
      CALL IPLOT
      OPEN(UNIT=23,DEVICE='DSK',FILE='FOR41.DAT')
      READ(23,*)(PIC(I),I=1,MX)
      PRINT 30
   30 FORMAT(1H1,///////,20X,'DIFFRACTION BLURRED IMAGE WITH
     WHITE GAUSSIAN NOISE N(0,0.1)')
      CALL IPLOT
      OPEN(UNIT=24,DEVICE='DSK',FILE='FOR48.DAT')
      READ(24,*)(PIC(I),I=1,NX)
      PRINT 40
   40 FORMAT(1H1,///////,20X,'RESTORED IMAGE BY FILTERING
     MAXIMUM ENTROPY PRINCIPLE')
      CALL PPLOT
      STOP
      END
C-----------------------------------------------------------
      SUBROUTINE PPLOT
      INTEGER NX,LX,MX,PIC
      REAL WB
      COMMON NX,LX,MX,PIC(32),WB(64)
      CALL SCALE(MX)
      PRINT 5
    5 FORMAT(1H ,////)
      DO 10 I=1,32
      DO 20 J=1,64
      WB(J)=' '
   20 CONTINUE
      DO 40 K=1,NX
      L=32-I
      IF(PIC(K).NE.L)GOTO 40
      WB(4*K)='*'
   40 CONTINUE
      PRINT 30,(WB(J),J=1,64)
   30 FORMAT(1H ,10X,'I',5X,64A1)
   10 CONTINUE
      PRINT 50
```

```
FORMAT(1H ,10X,'!',/,10X,'-----------------------------------------
-----------------------------------')
RETURN
END
-------------------------------------------------------------------
SUBROUTINE IPLOT
INTEGER NX,LX,MX,PIC
REAL WB
COMMON NX,LX,MX,PIC(32),WB(64)
CALL SCALE(MX)
PRINT 5
FORMAT(1H ,////)
DO 10 I=1,32
DO 20 J=1,64
WB(J)=' '
CONTINUE
DO 40 K=1,MX
L=32-I
IF(PIC(K).NE.L)GOTO 40
WB(2*K)='*'
CONTINUE
PRINT 30,(WB(J),J=1,64)
FORMAT(1H ,10X,'!',5X,64A1)
CONTINUE
PRINT 50
FORMAT(1H0,10X,'!',/,10X,'-----------------------------------------
-----------------------------------')
RETURN
END
-------------------------------------------------------------------
SUBROUTINE SCALE(NO)
INTEGER NO,NX,LX,MX,PIC,MIN,MAX
REAL WB,SUM
COMMON NX,LX,MX,PIC(32),WB(64)
MAX=0
MIN=0
DO 10 I=1,NO
IF(PIC(I).GE.MIN)GOTO 20
MIN=PIC(I)
IF(PIC(I).LE.MAX)GOTO 30
MAX=PIC(I)
CONTINUE
CONTINUE
IF(MIN.GE.0)GOTO 40
DO 50 I=1,NO
PIC(I)=PIC(I)-MIN
CONTINUE
MAX=MAX-MIN
IF(MAX.LE.31)GOTO 60
DO 70 I=1,NO
SUM=PIC(I)*31.0/MAX
PIC(I)=IFIX(SUM)
CONTINUE
CONTINUE
RETURN
END
```

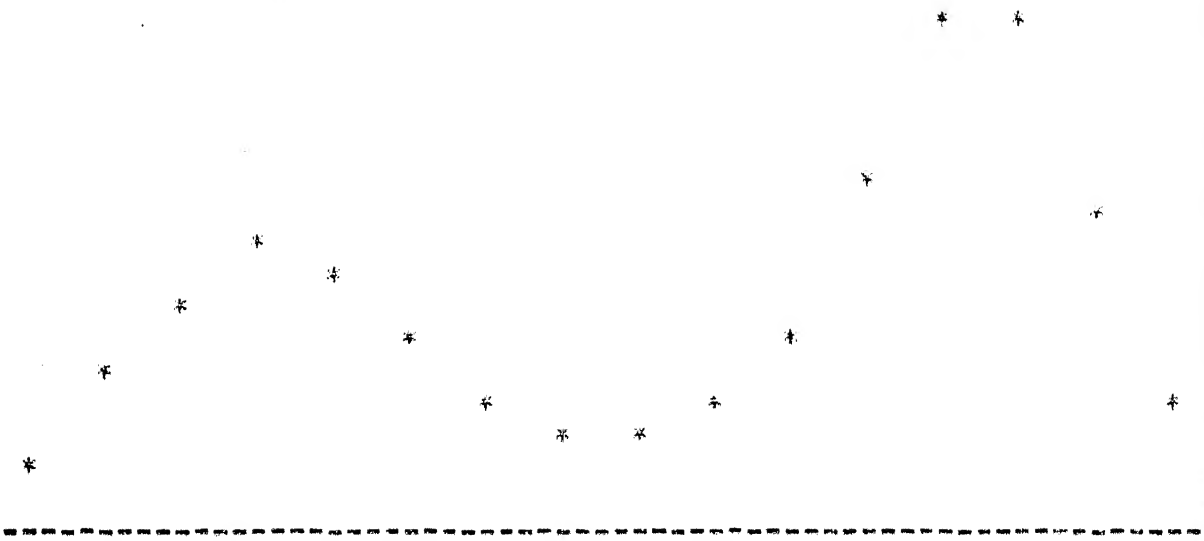
THE OBJECT IMAGE OF SIZE 16

DIFFRACTION BLURRED IMAGE WITH WHITE GAUSSIAN NOISE N(0,4.1)

SNR=40 DB

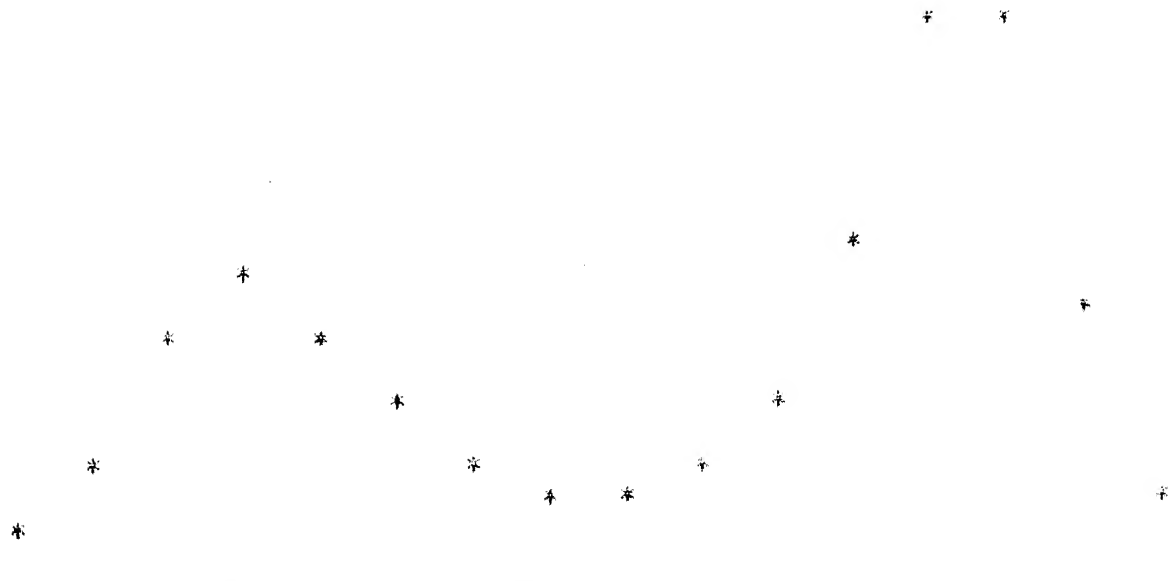
RESTORED IMAGE BY FRIEDEN'S MAXIMUM ENTROPY PRINCIPLE

WITH A=0.2,N0=10.0



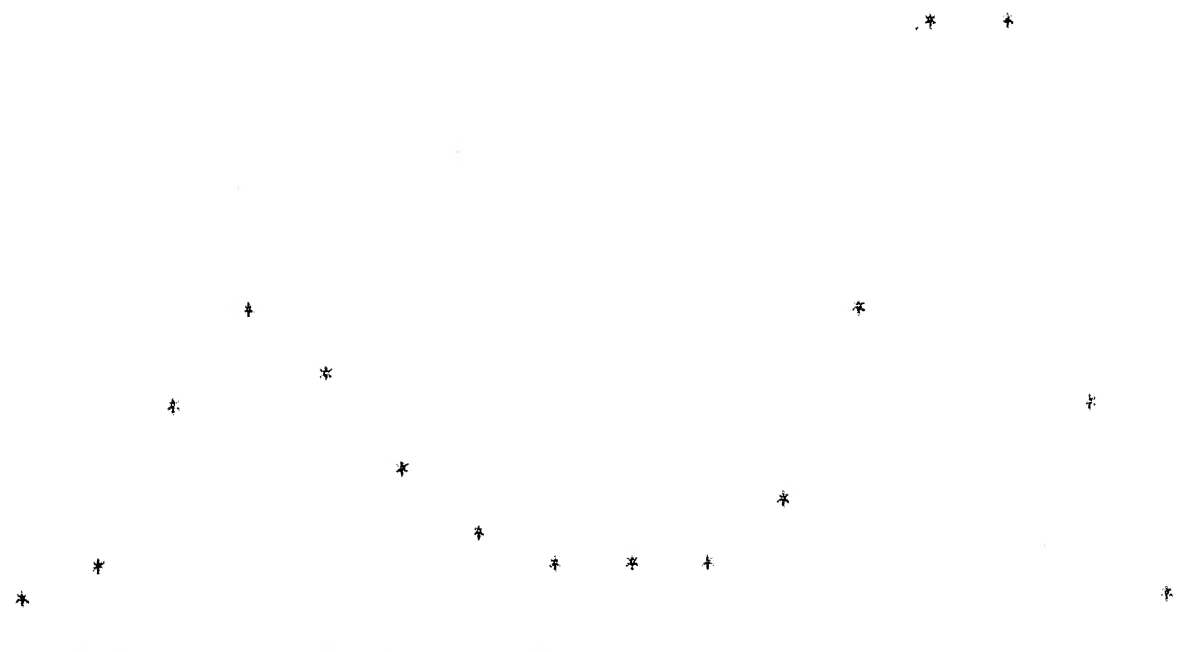
RESTORED IMAGE BY FRIEDENS MAXIMUM ENTROPY PRINCIPLE

WITH B=0.2,$\bar{R0}$=26



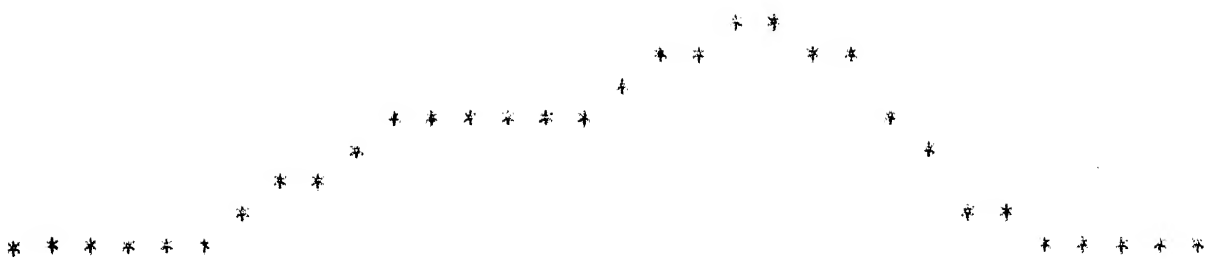
RESTORED IMAGE BY FRIEDENS MAXIMUM ENTROPY PRINCIPLE

WITH B=0.2,K0=40



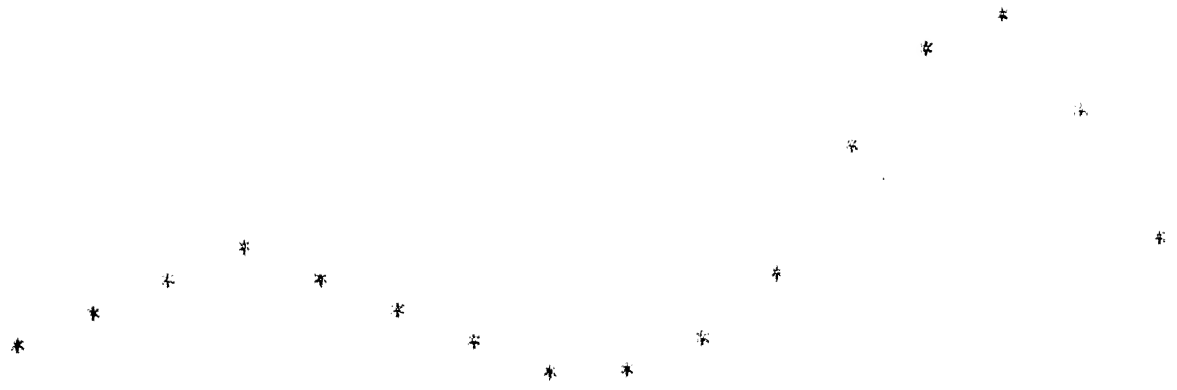
IFFRACTION BLURRED IMAGE WITH WHITE GAUSSIAN NOISE N(0,0.1)

SNR=35 dB



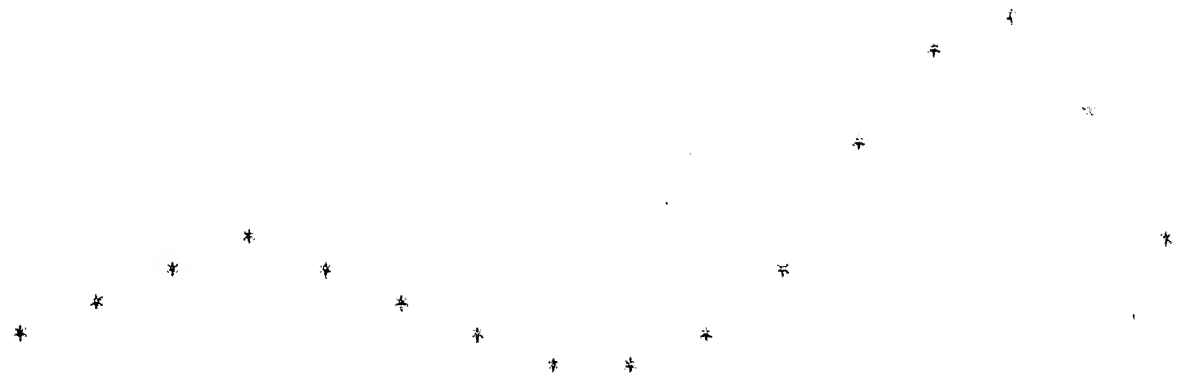
RESTORED IMAGE BY FRIEDENS MAXIMUM ENTROPY PRINCIPLE

WITH B=0.4,D0=10



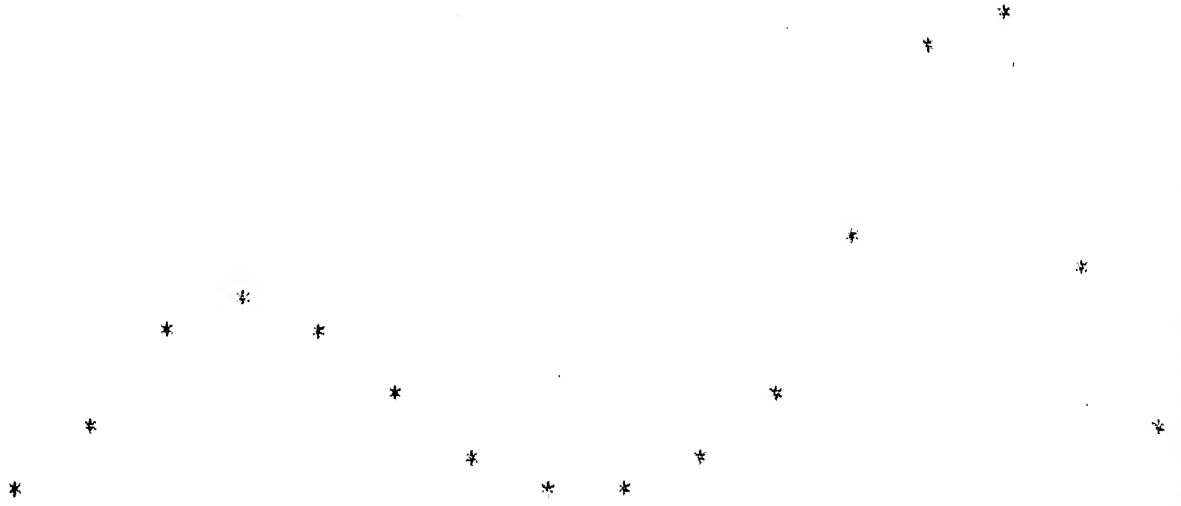
RESTORED IMAGE BY FRIEDEN'S MAXIMUM ENTROPY PRINCIPLE

WITH b=0.1,P0=20.0

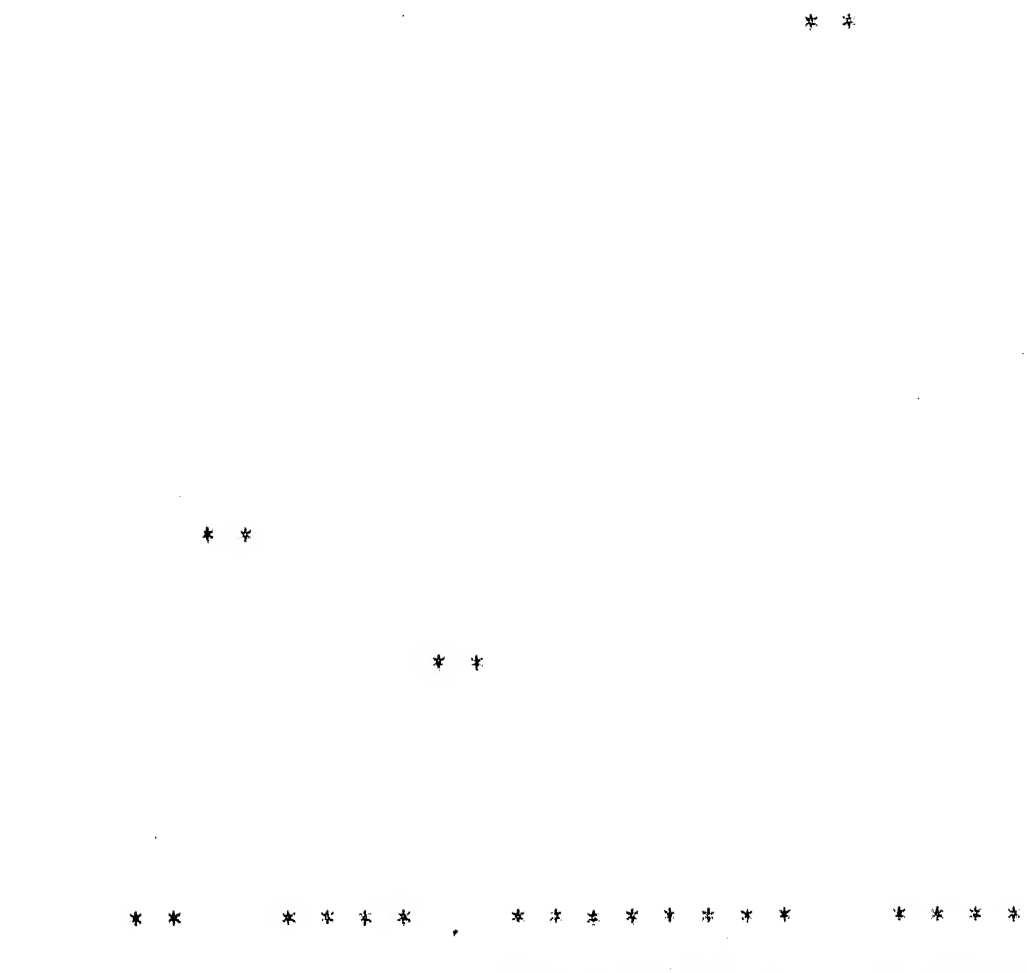


RESTORED IMAGE BY FRIEDEN'S MAXIMUM ENTROPY PRINCIPLE

WITH R=0.4,K0=40.0

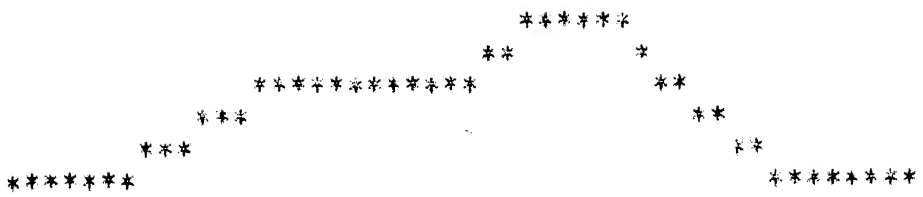


THE OBJECT IMAGE OF SIZE 24



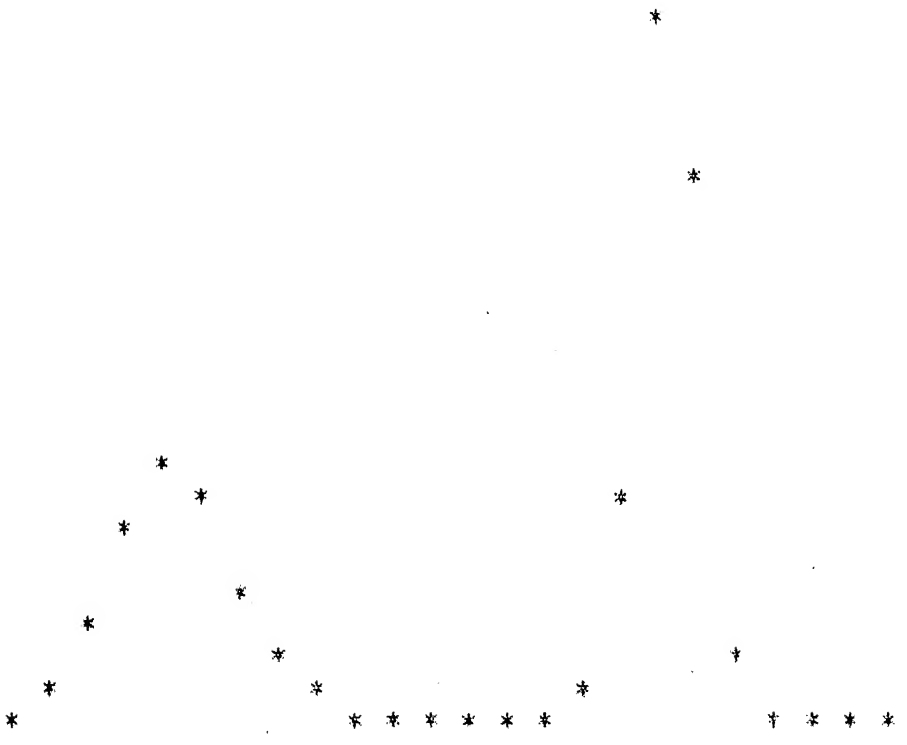
DIFFRACTION BLURRED IMAGE WITH WHITE GAUSSIAN NOISE N(0,0.1)

S/R=40 DB



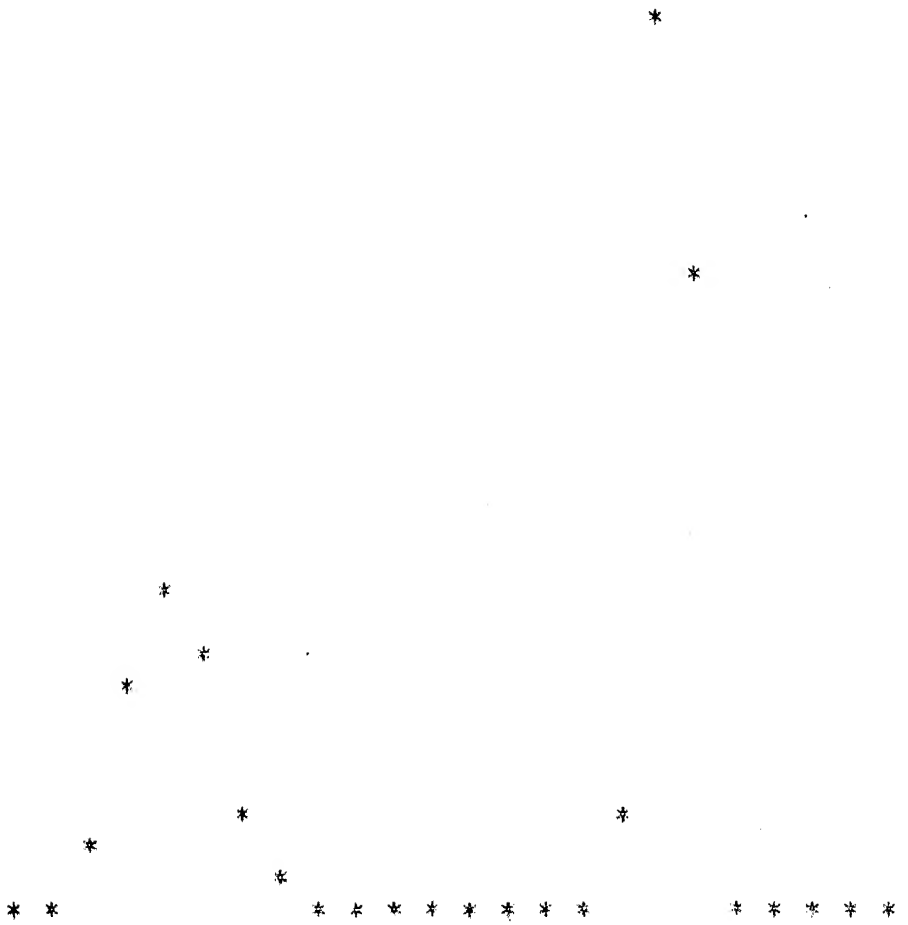
RESTORED IMAGE BY FRIEDENS MAXIMUM ENTROPY PRINCIPLE

WITH B=0.2,RO=10.0



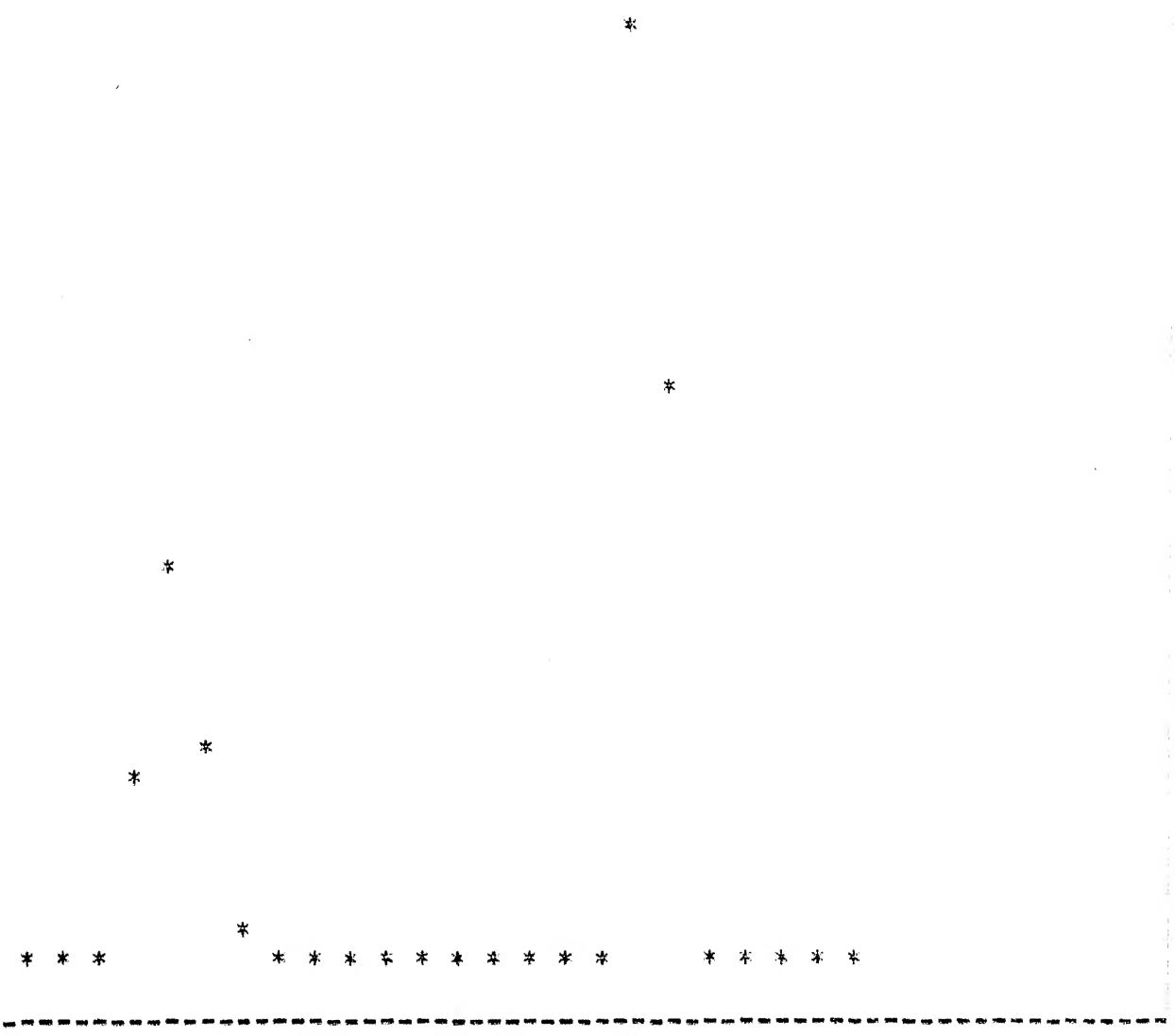
RESTORED IMAGE BY FRIEDENS MAXIMUM ENTROPY PRINCIPLE

WITH A=0.2,B0=20.3



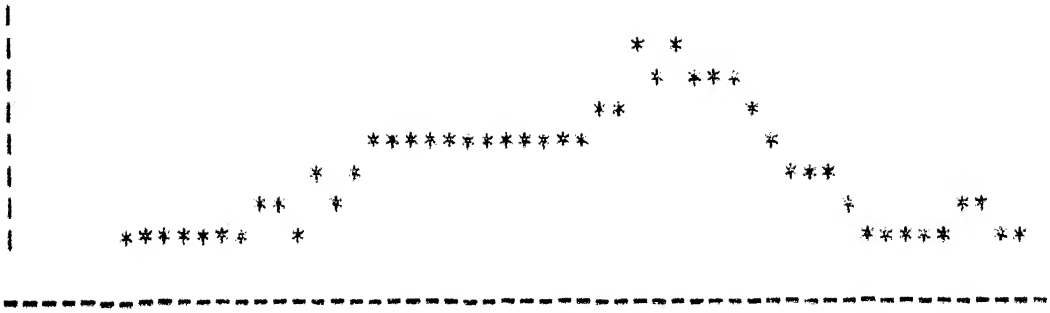
RESTORED IMAGE BY FRIEDENS MAXIMUM ENTROPY PRINCIPLE

WITH B=0.2,R0=40.0



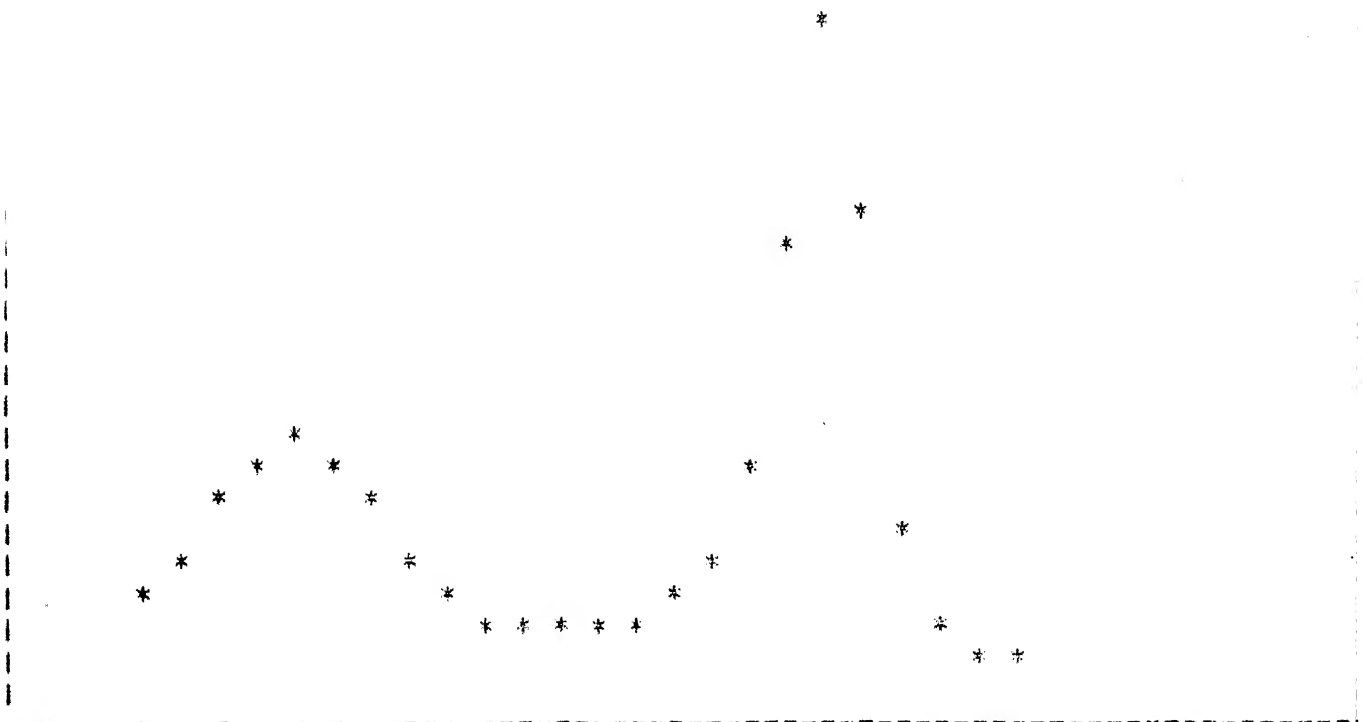
DIFFRACTION BLURRED IMAGE WITH WHITE GAUSSIAN NOISE N(0,0.5)

SNR=27 DB

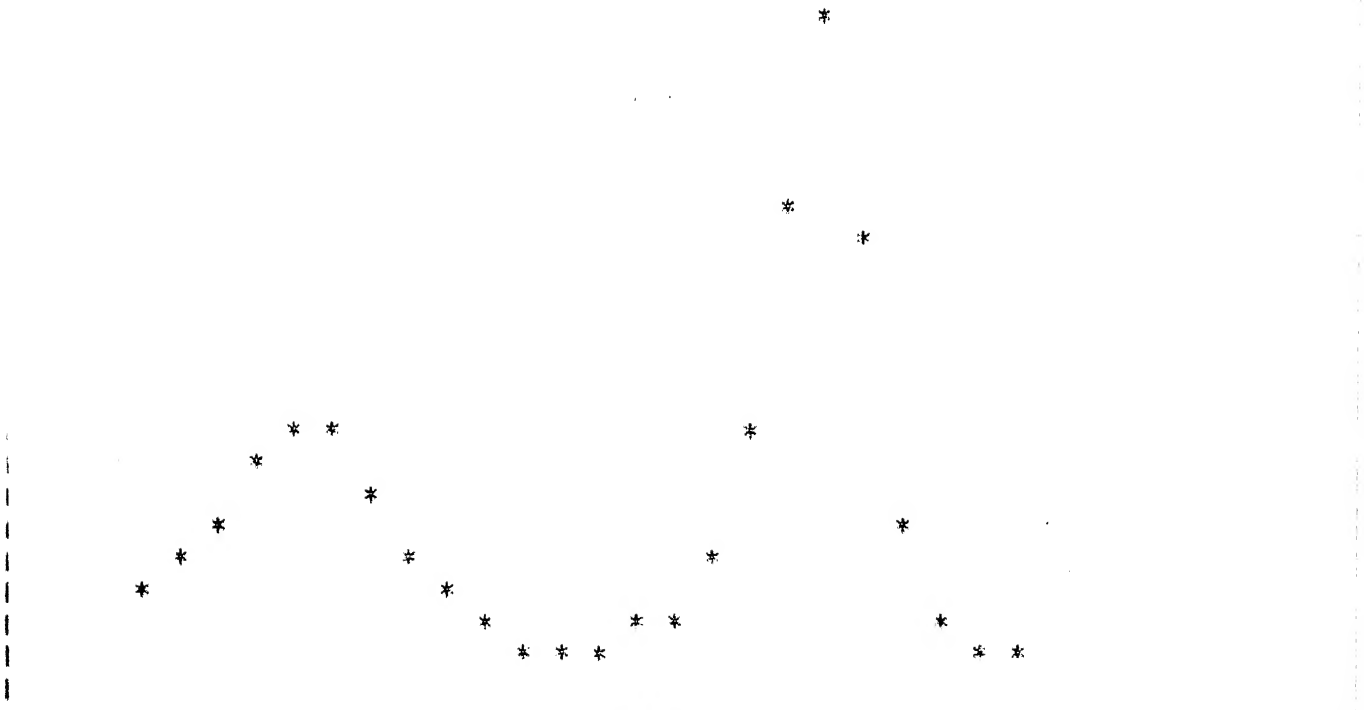


RESTORED IMAGE BY FRIEDENS MAXIMUM ENTROPY PRINCIPLE

WITH R=1.0,RÔ=10.0

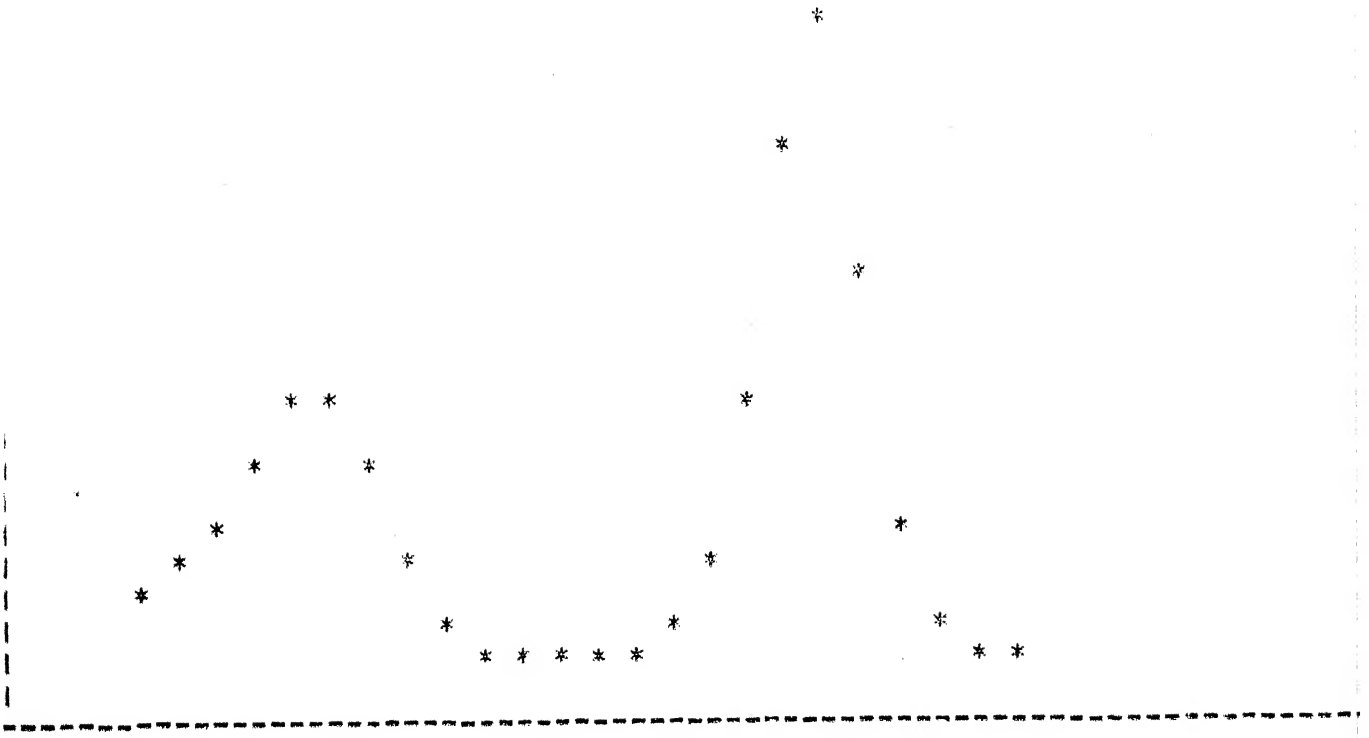


RESTORED IMAGE BY FRIEDENS MAXIMUM ENTROPY PRINCIPLE
WITH B=1.0,R0=20.0

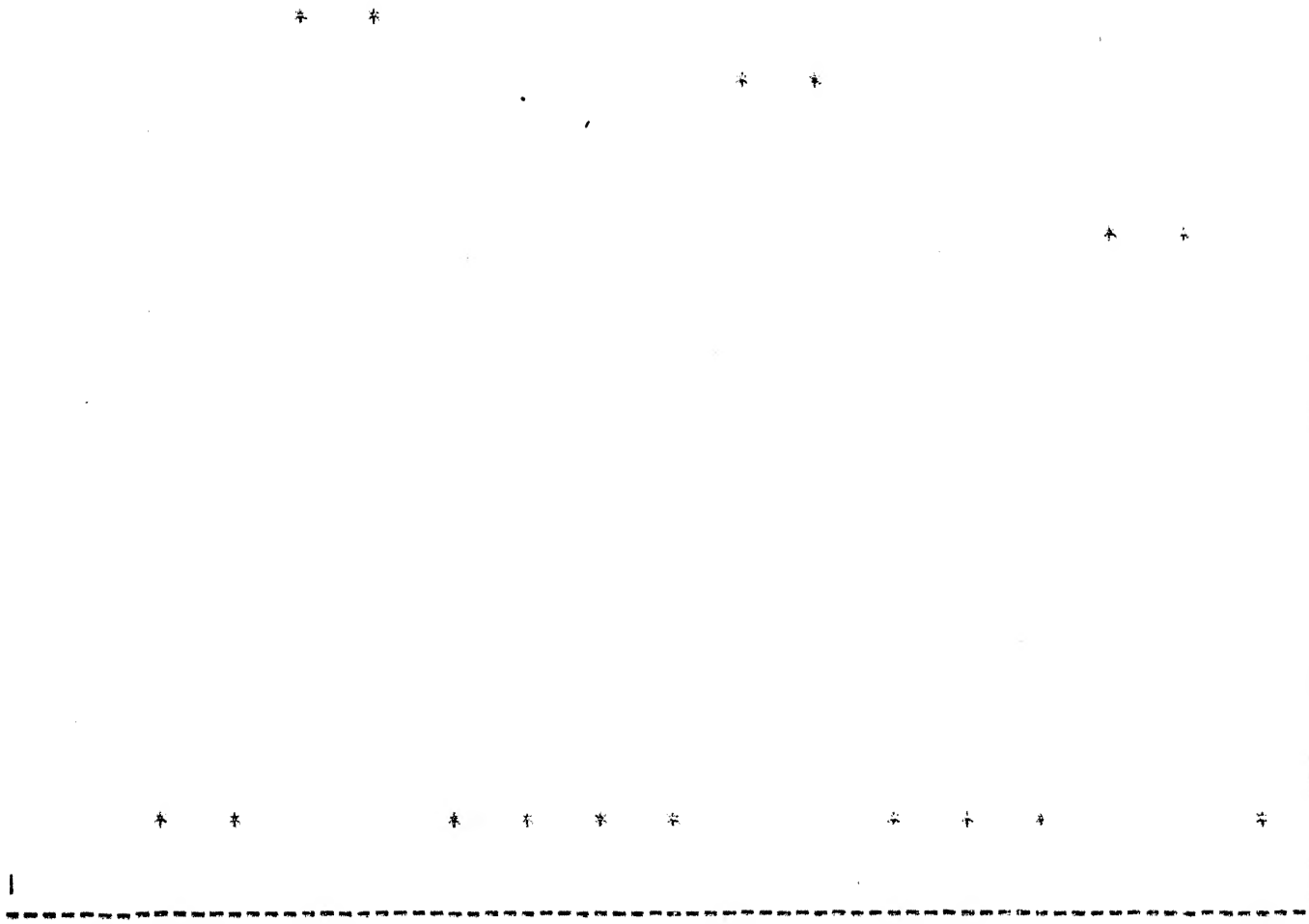


RESTORED IMAGE BY FRIEDEN'S MAXIMUM ENTROPY PRINCIPLE

WITH B=1.0,R0=20.0

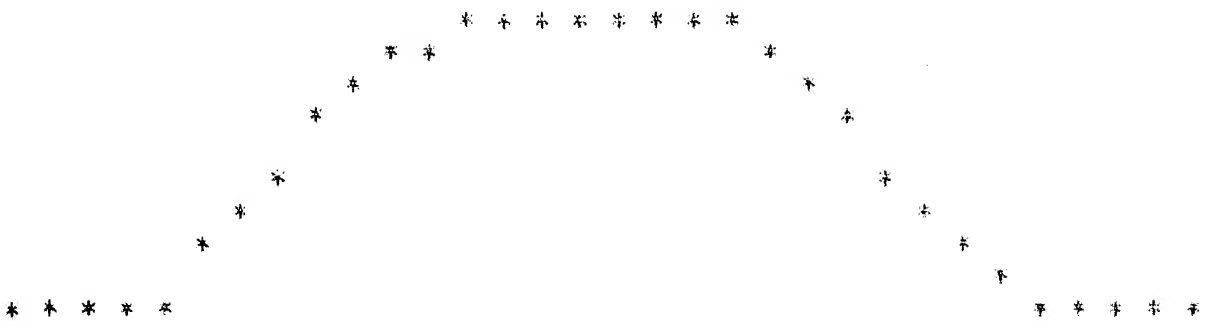


THE OBJECT IMAGE OF SIZE 16



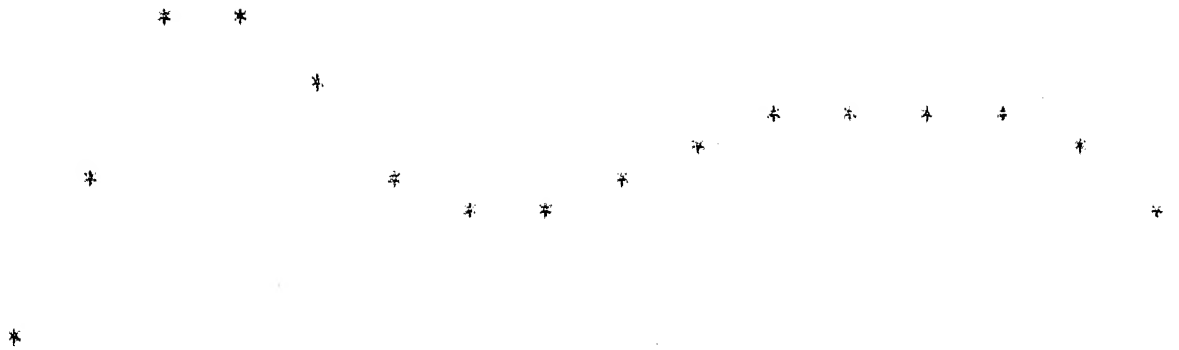
DIFFRACTION BLURRED IMAGE WITH WHITE GAUSSIAN NOISE N(0,0.1)

SNR=42 DB



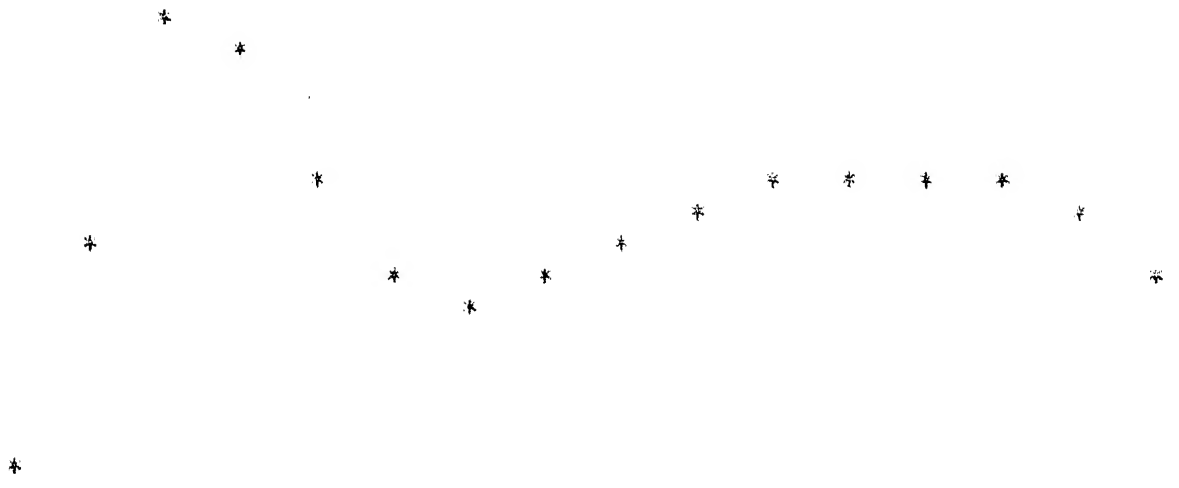
RESTORED IMAGE BY FRIEDENS MAXIMUM ENTROPY PRINCIPLE

WITH B=0.2,P0=10



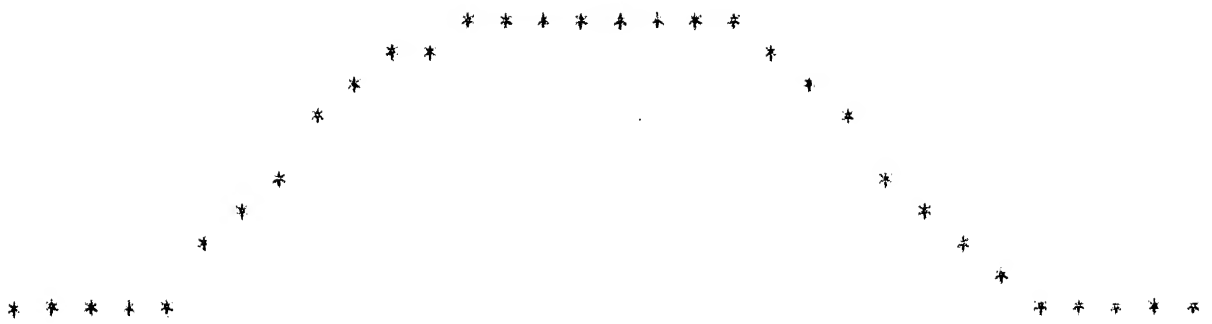
RESTORED IMAGE BY FRIEDENS ARATIOS ENTROPY PRINCIPLE

WITH a=0.2,K0=20



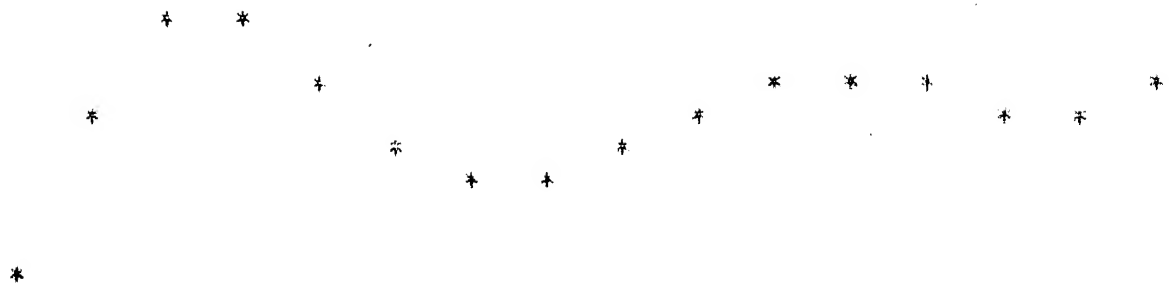
DIFFRACTION BLURRED IMAGE WITH WHITE GAUSSIAN NOISE N(0,0.2)

SNR=37 DB



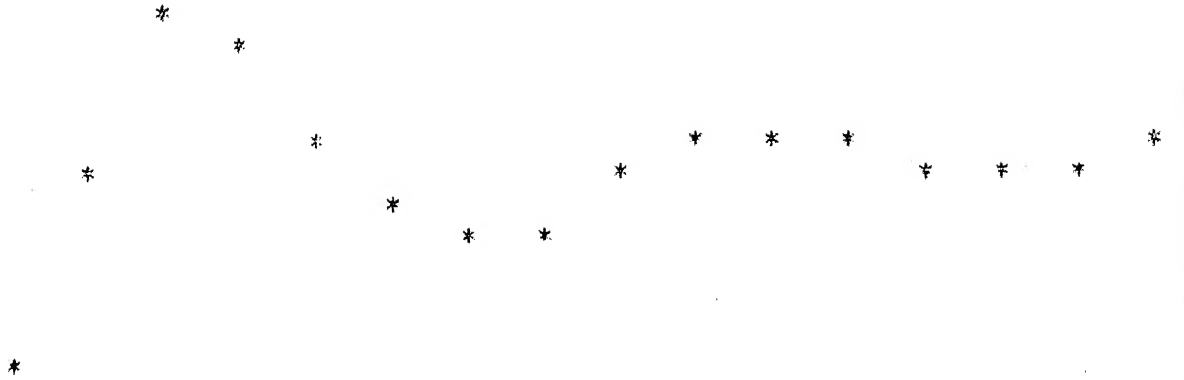
RESTORED IMAGE BY FRIEDENS MAXIMUM ENTROPY PRINCIPLE

WITH b=0.4,K0=10



RESTORED IMAGE BY FRIEDENS MAXIMUM ENTROPY PRINCIPLE

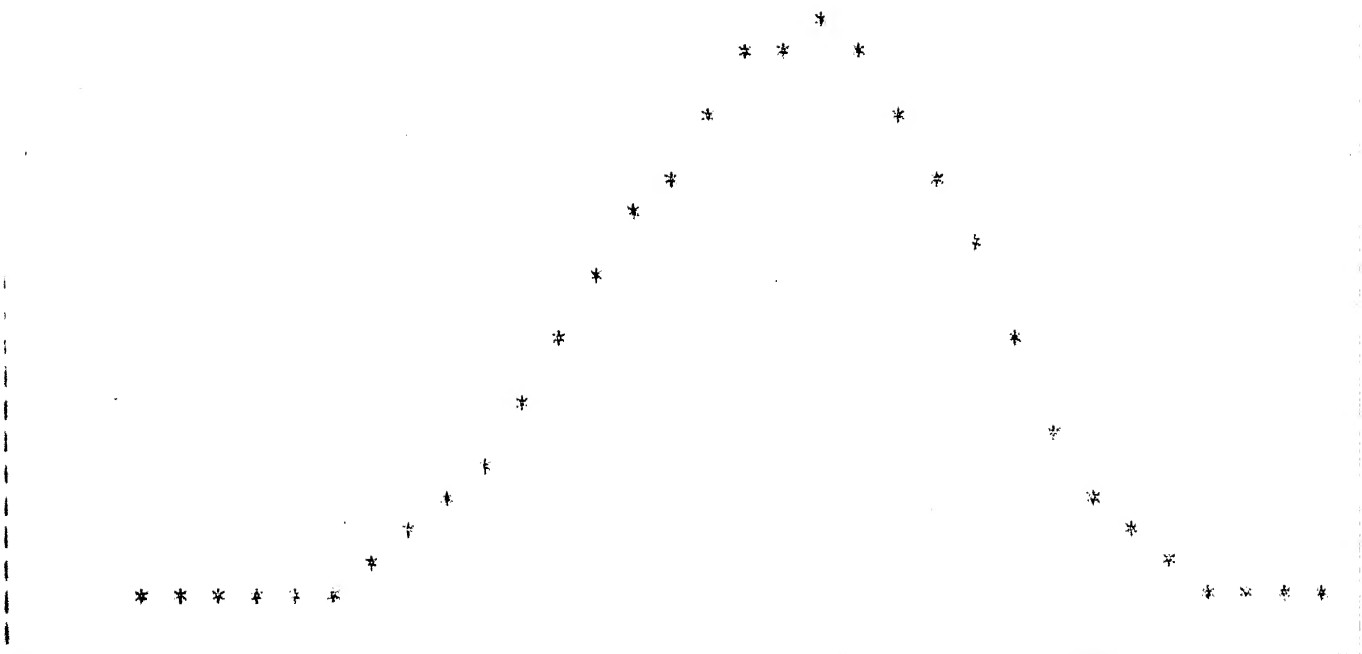
WITH B=0.4,R0=20



THE OBJECT IMAGE OF SIZE 16

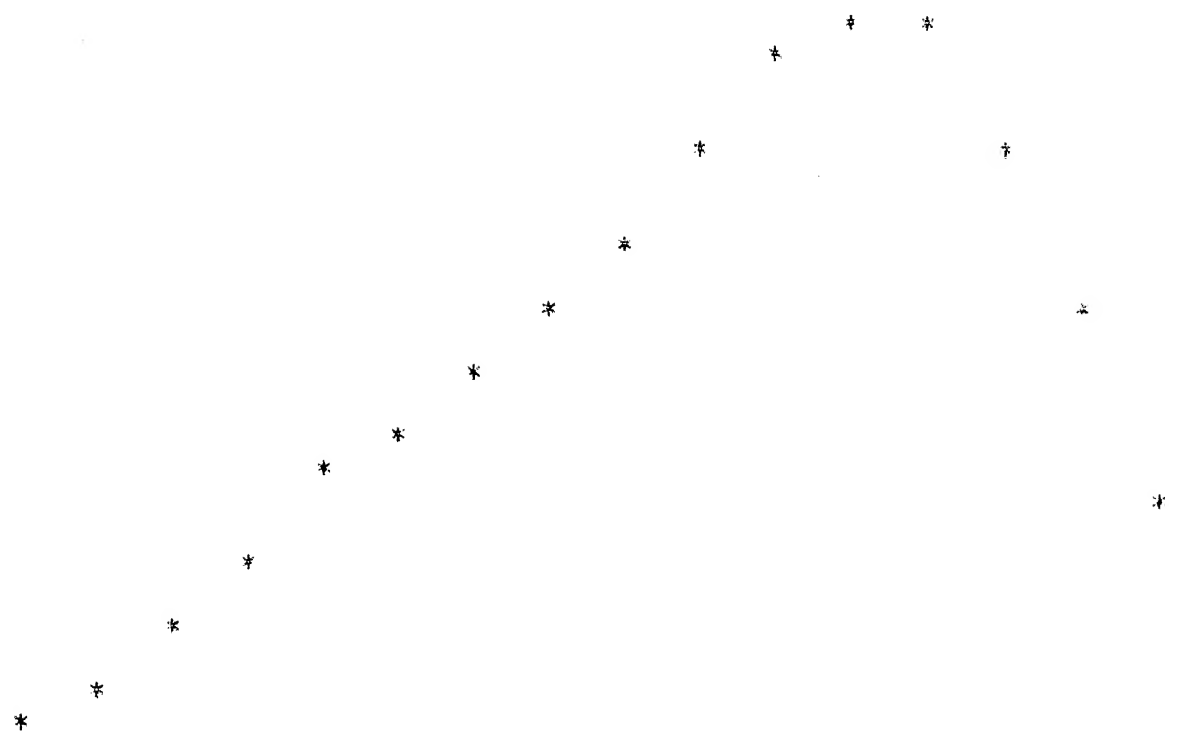
DIFFRACTION BLURRED IMAGE WITH WHITE GAUSSIAN NOISE N(0,0.1)

S/R=43 DB



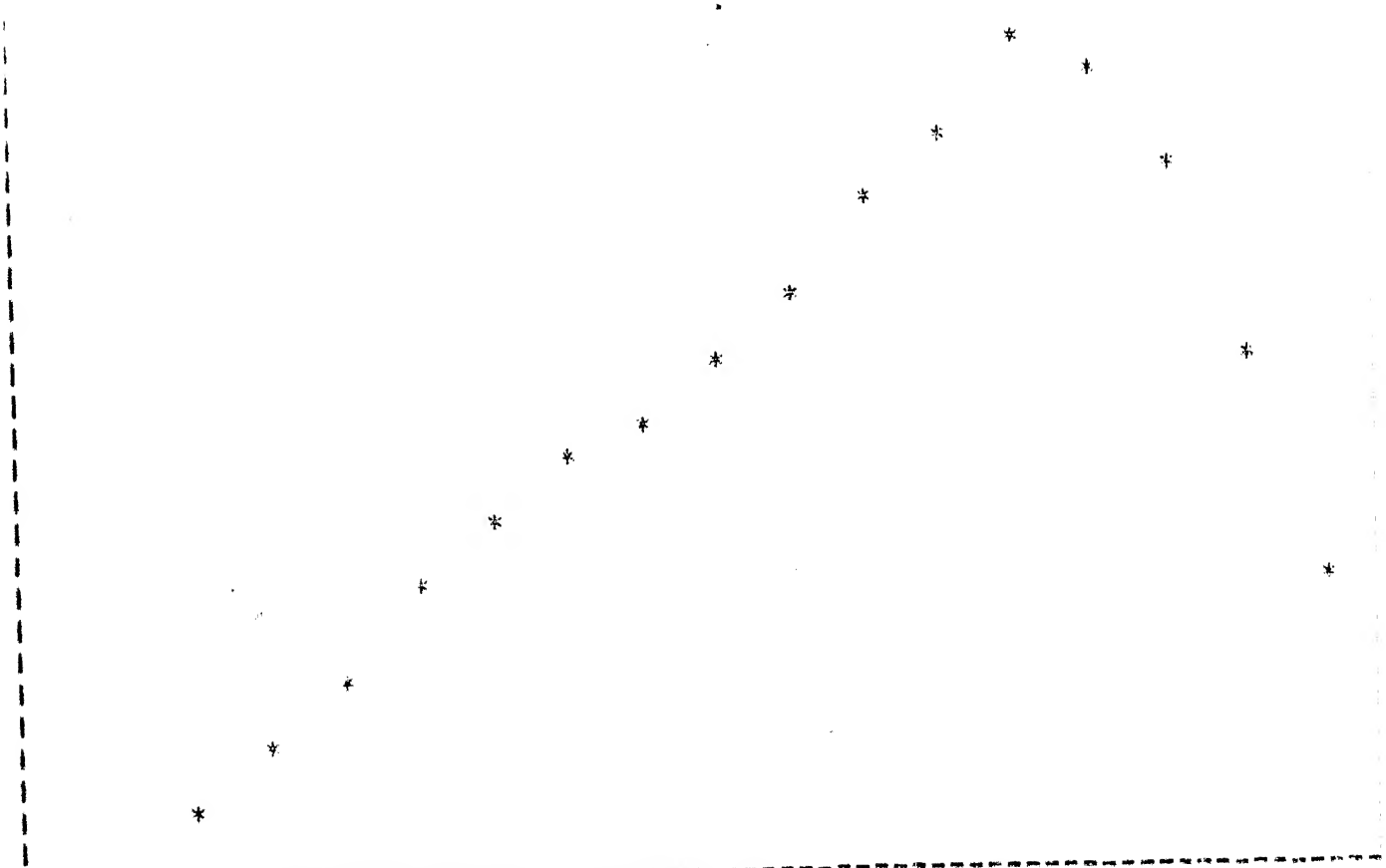
RESTORED IMAGE BY FRIEDENS MAXIMUM ENTROPY PRINCIPLE

WITH B=0.2,R0=10.0



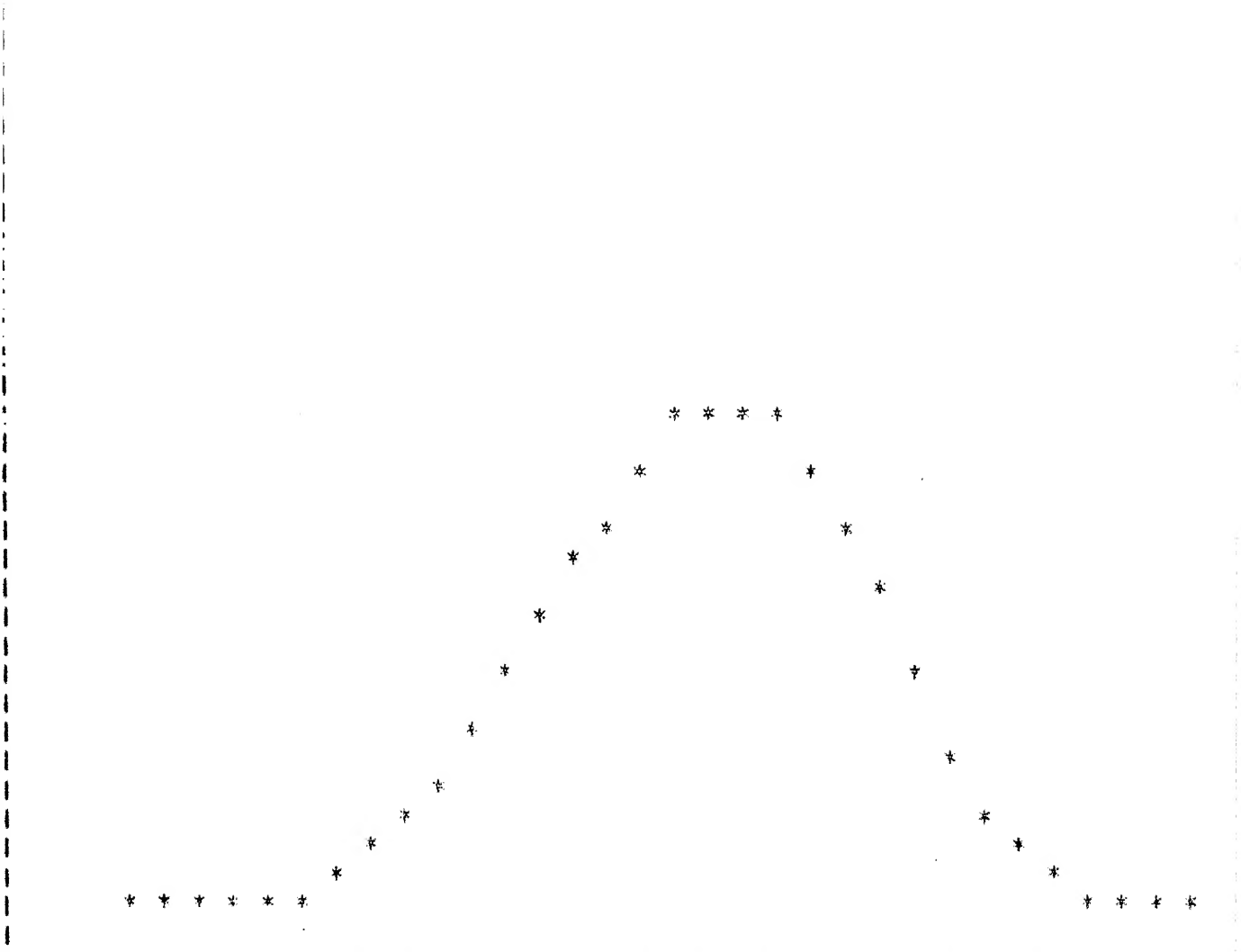
RESTORED IMAGE BY FRIEDENS MAXIMUM ENTROPY PRINCIPLE

WITH $\alpha$=0.2, $K\Omega$=20



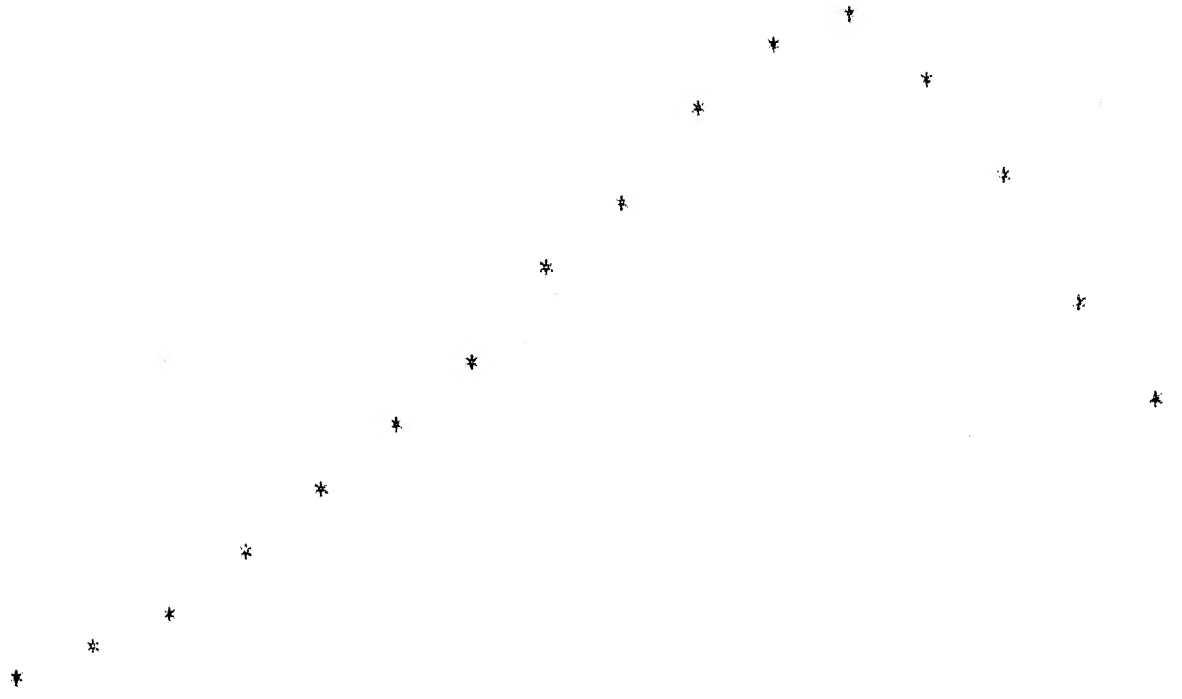
DIFFRACTION BLURRED IMAGE WITH WHITE GAUSSIAN NOISE N(0,0.2)

S/P=37 DB



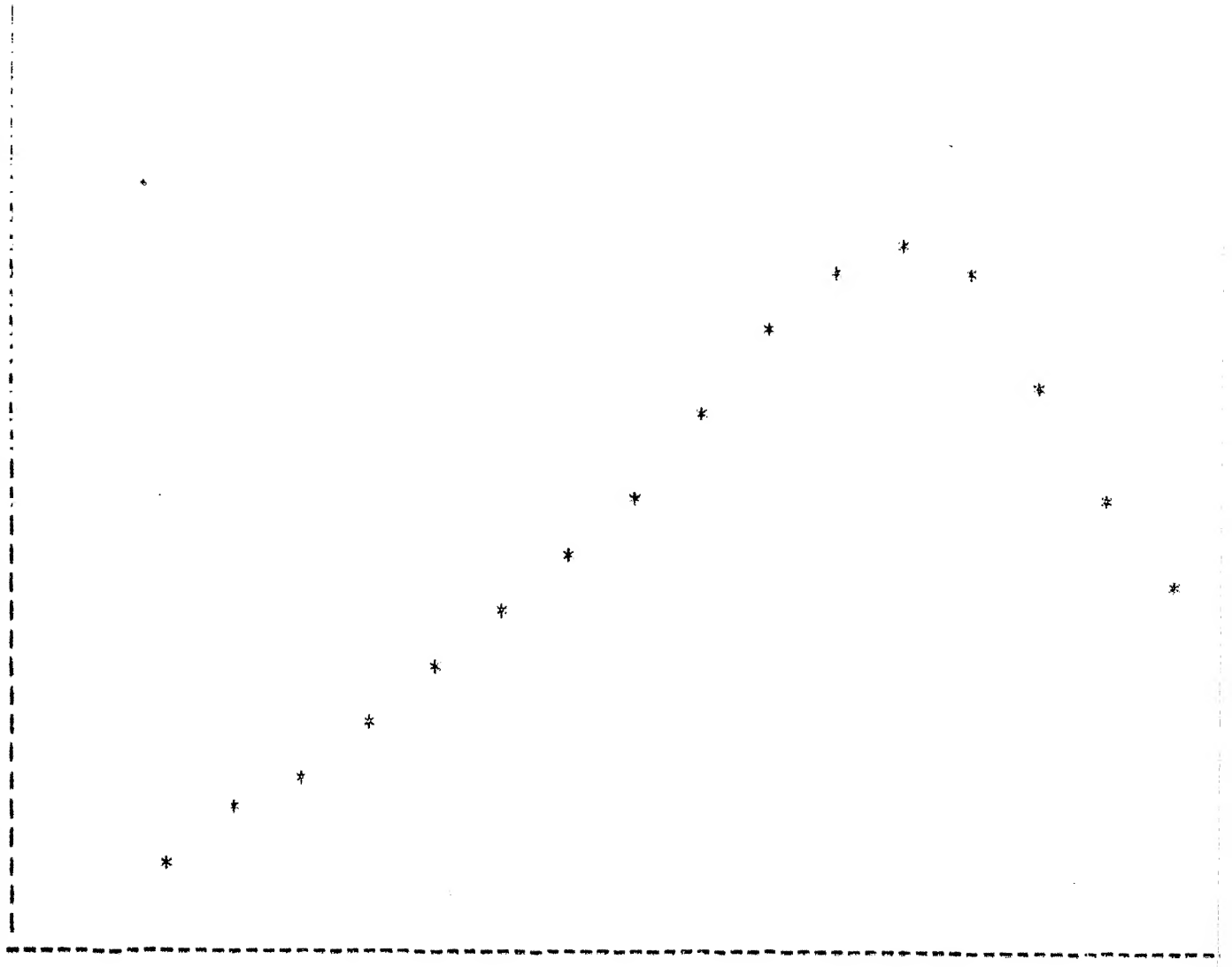
RESTORED IMAGE BY EXTENDED MAXIMUM ENTROPY PRINCIPLE

WITH R=0.4,RO=10.0

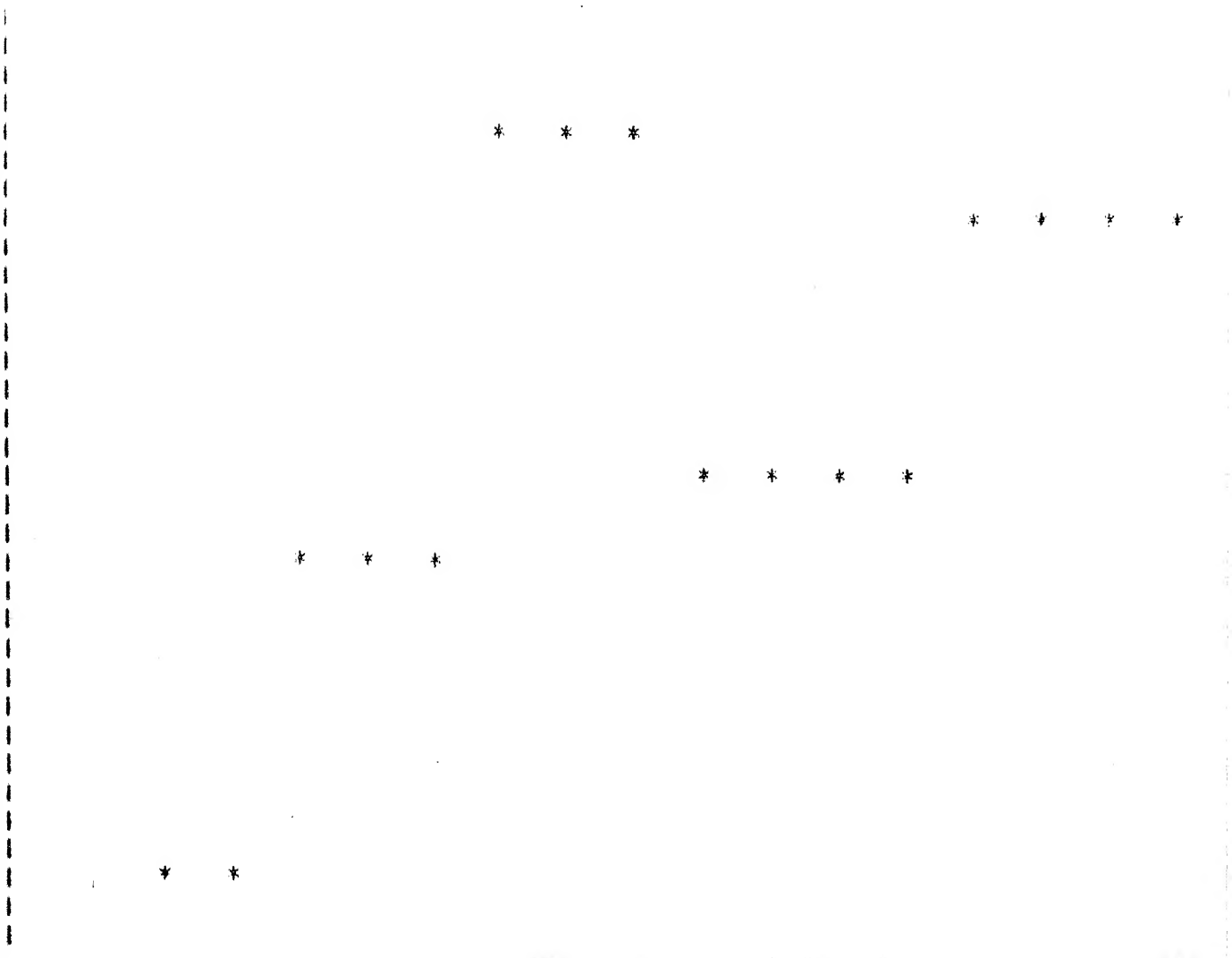


RESTORED IMAGE BY FRIEDENS MAXIMUM ENTROPY PRINCIPLE

WITH B=0.4,KO=20

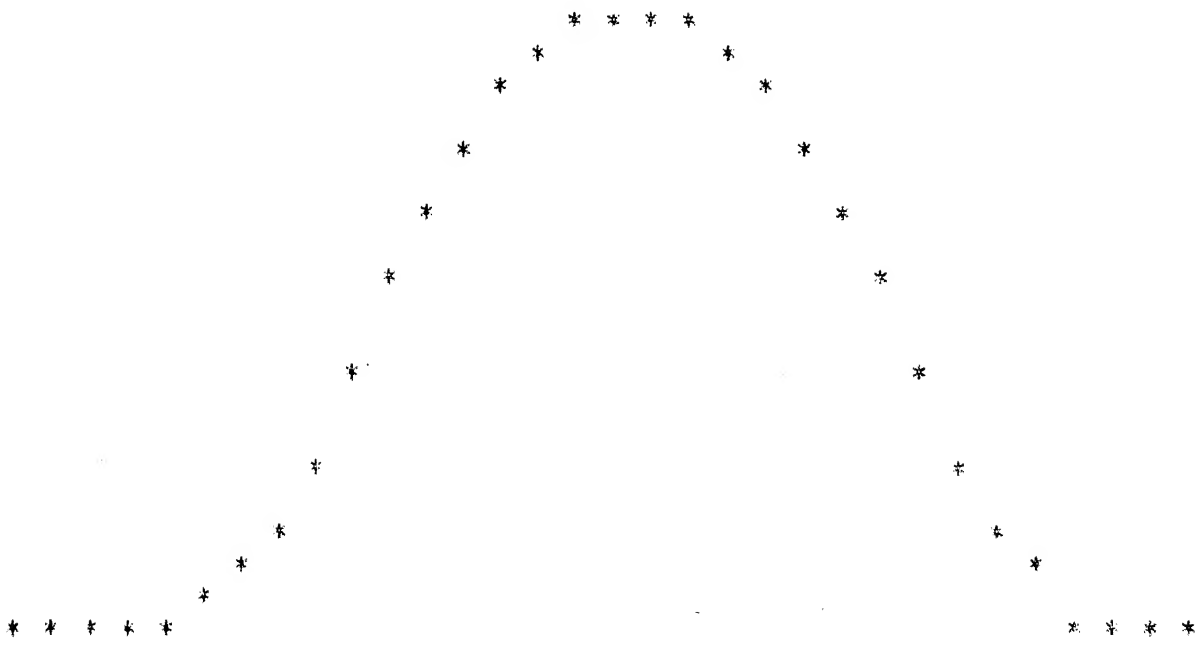


THE OBJECT IMAGE OF SIZE 16



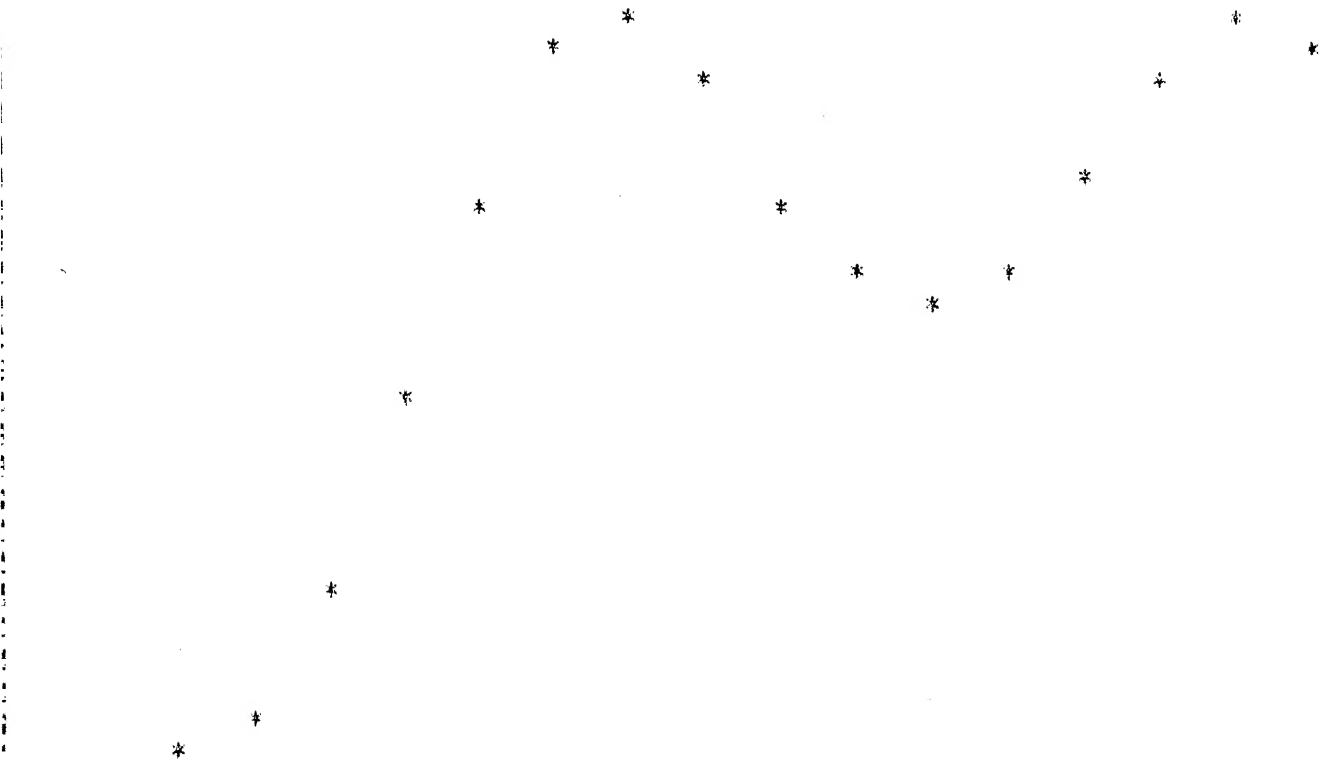
DIFFRACTION BLURRED IMAGE WITH WHITE GAUSSIAN NOISE N(0,0.1)

SNR=45 DB



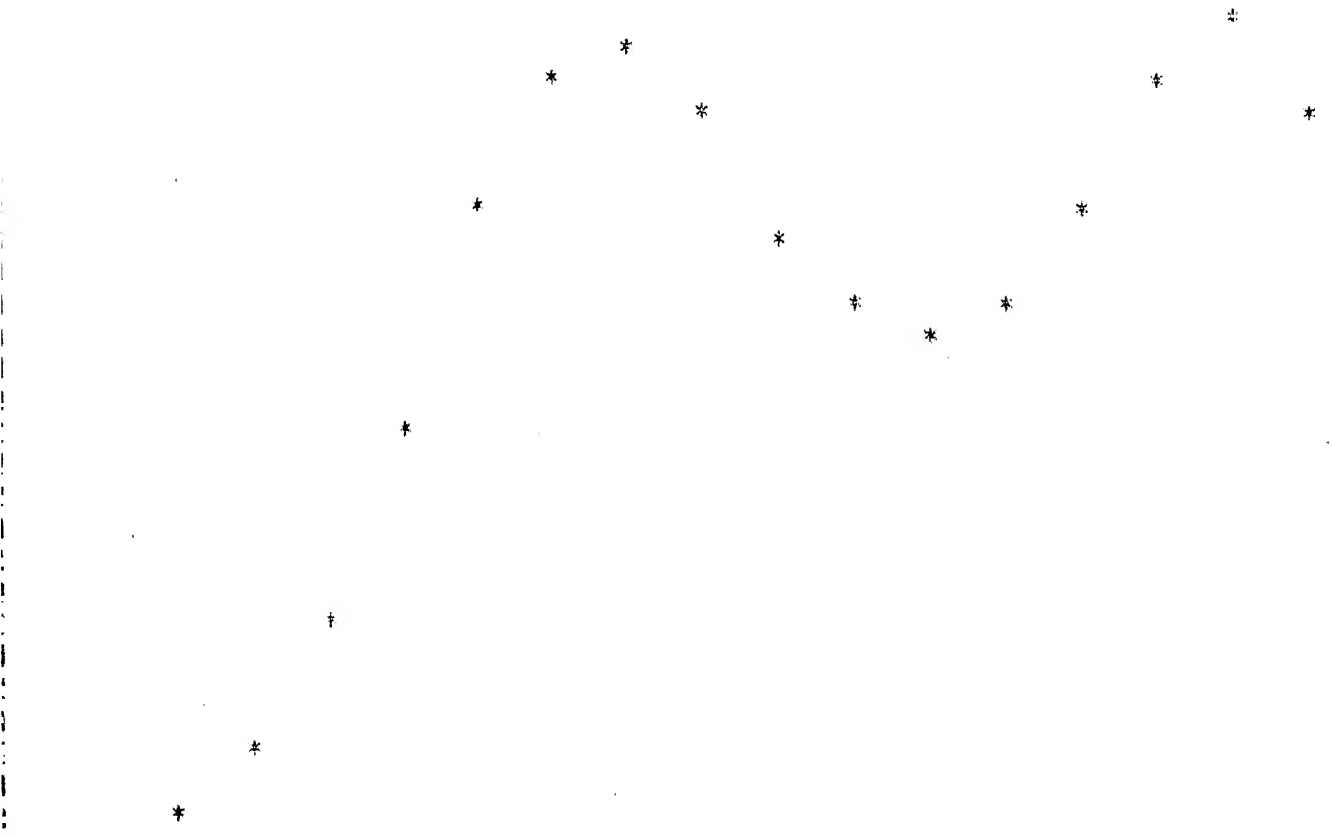
RESTORED IMAGE BY FRIEDENS MAXIMUM ENTROPY PRINCIPLE

WITH B=0.2,R0=10.0



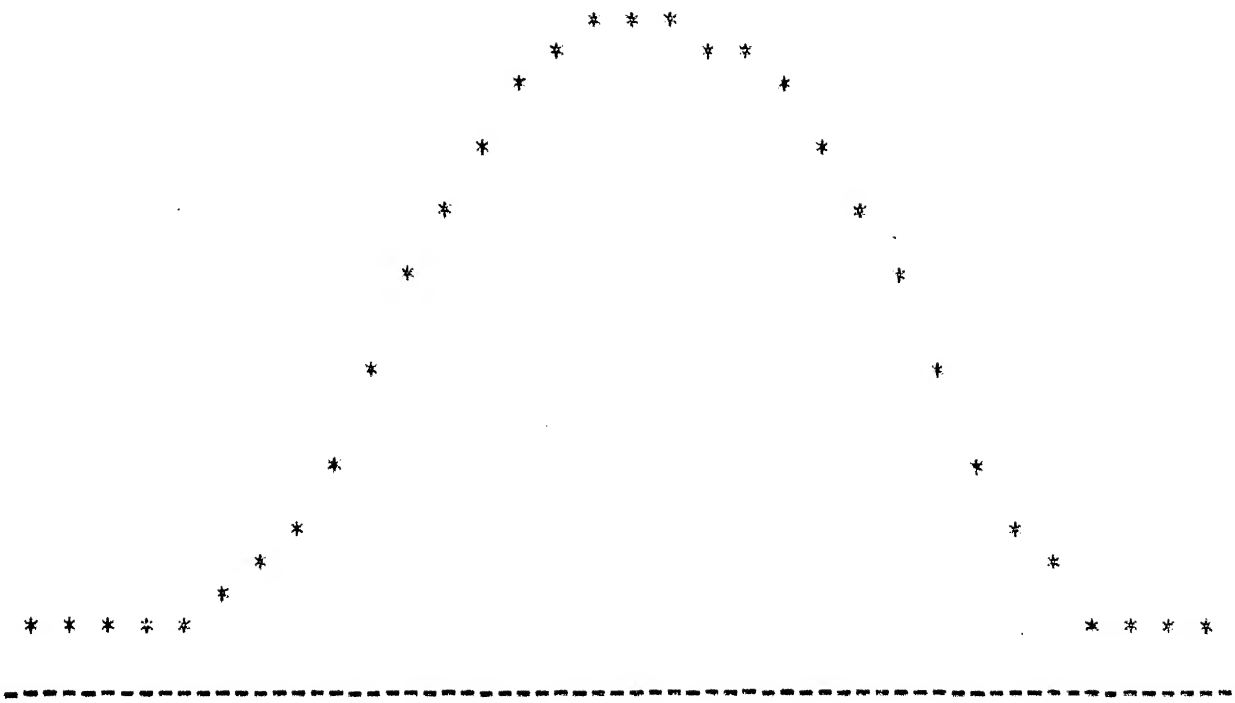
RESTORED IMAGE BY FRIEDENS MAXIMUM ENTROPY PRINCIPLE

WITH $\sigma^2=0.2, \alpha=20.0$



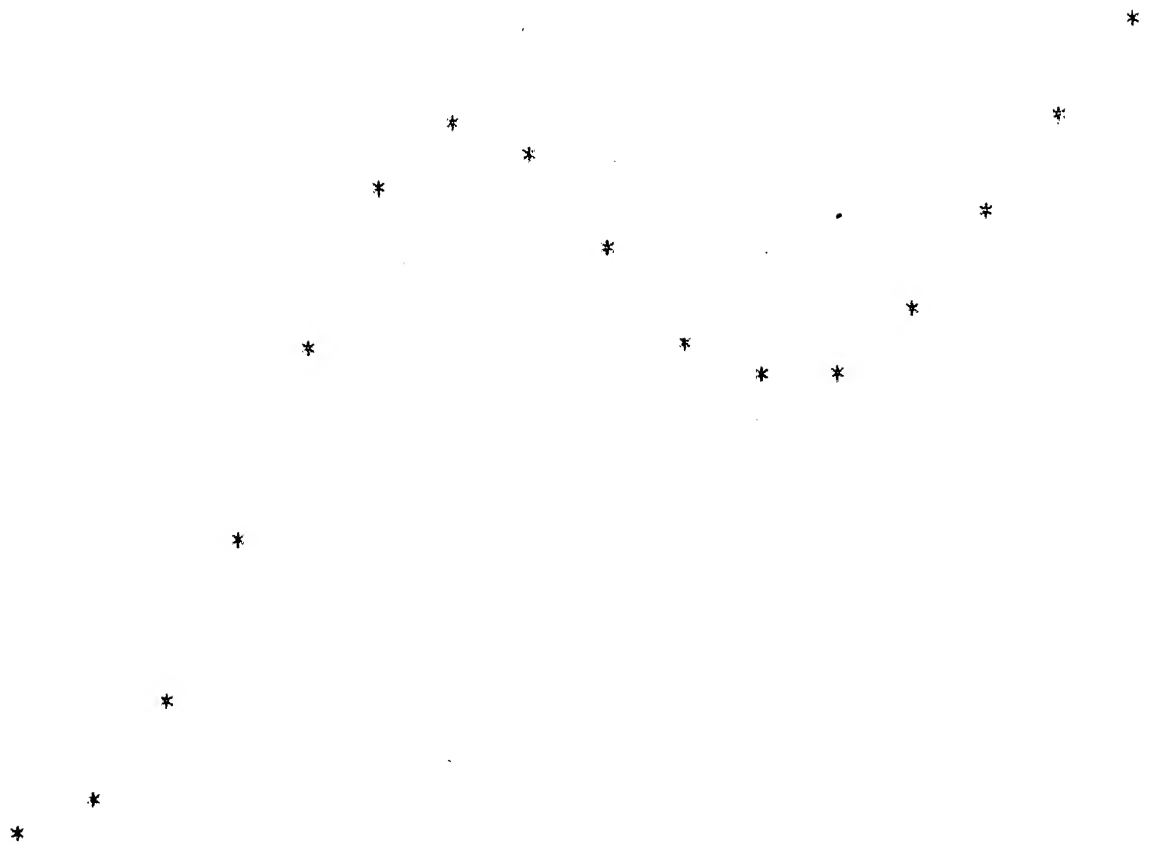
DIFFRACTION BLURRED IMAGE WITH WHITE GAUSSIAN NOISE N(0,0.2)

SNR=39 DB



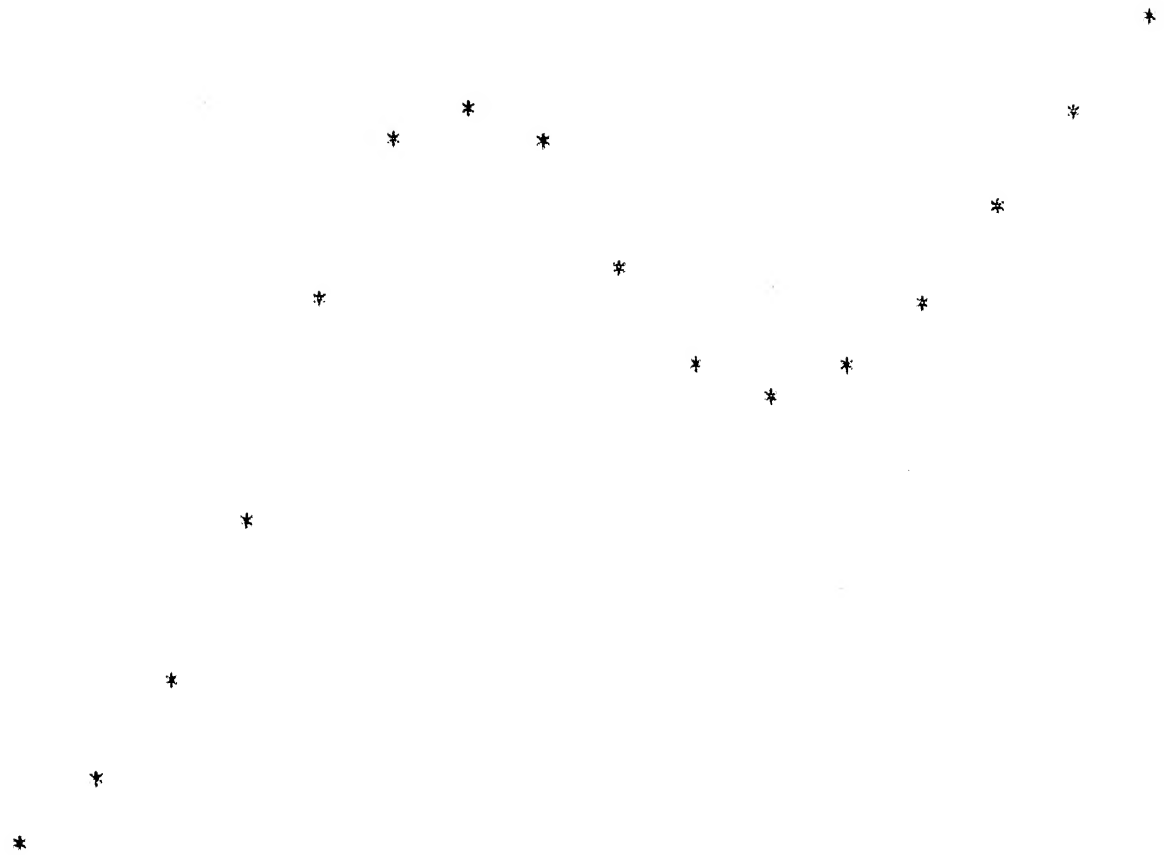
RESTORED IMAGE BY FRIEDENS MAXIMUM ENTROPY PRINCIPLE

WITH B=0.4,K0=10.0



RESTORED IMAGE BY FRIEDENS MAXIMUM ENTROPY PRINCIPLE

WITH B=0.4,R0=20.0



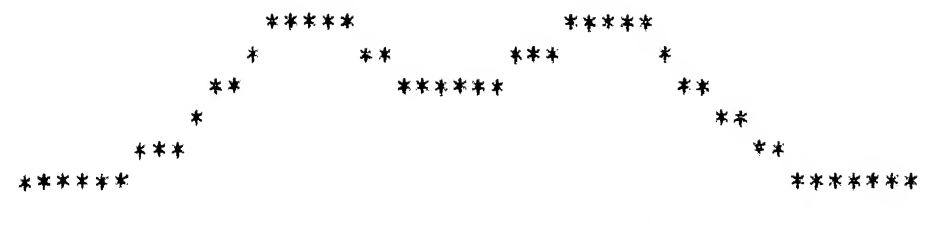
RESTORED IMAGE BY FRIEDENS MAXIMUM ENTROPY PRINCIPLE

WITH B=0.4,R0=40.0

THE OBJECT IMAGE OF SIZE 24

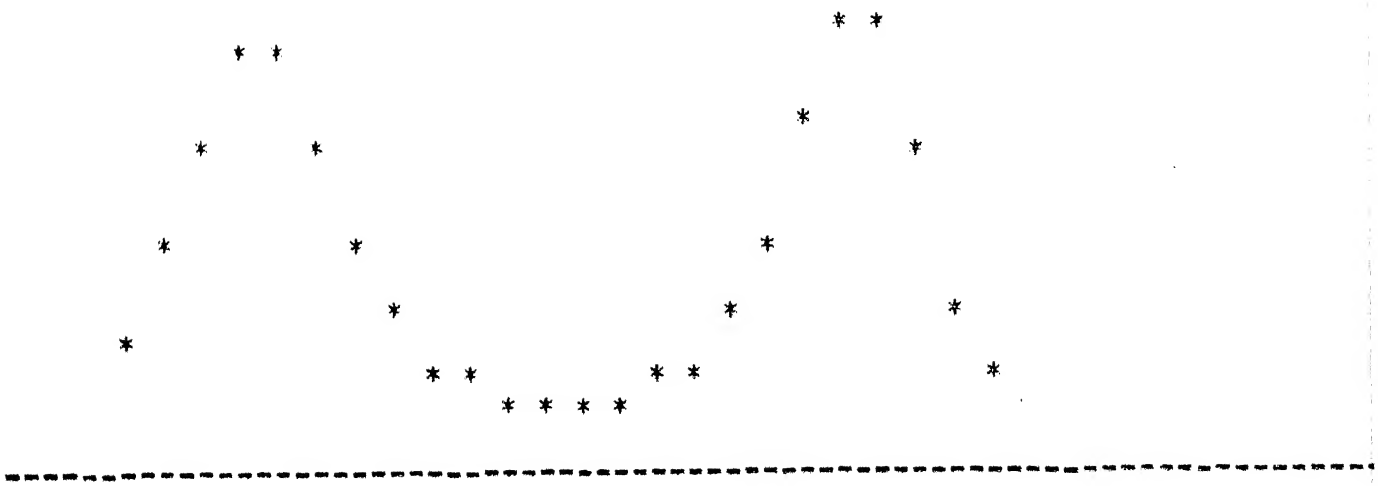
DIFFRACTION BLURRED IMAGE WITH WHITE GAUSSIAN NOISE N(0,0.1)

SNR=30 DB



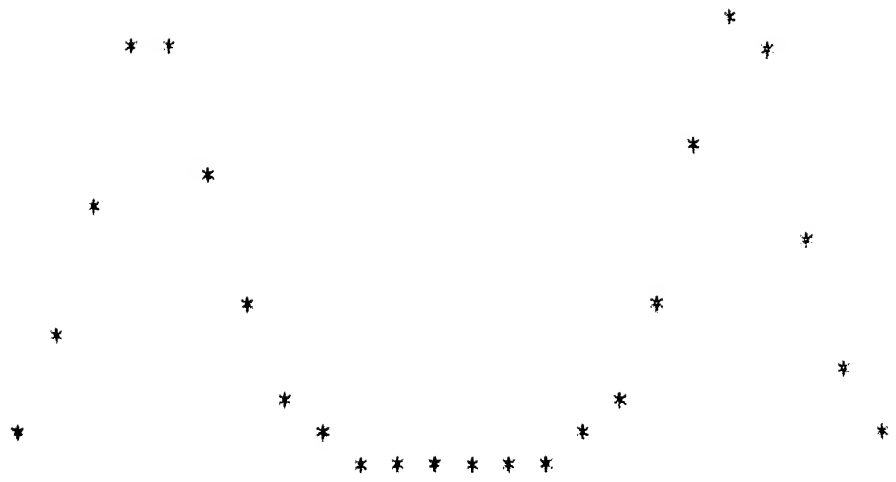
RESTORED IMAGE BY FRIEDENS MAXIMUM ENTROPY PRINCIPLE

WITH B=0.2,R0=10.0



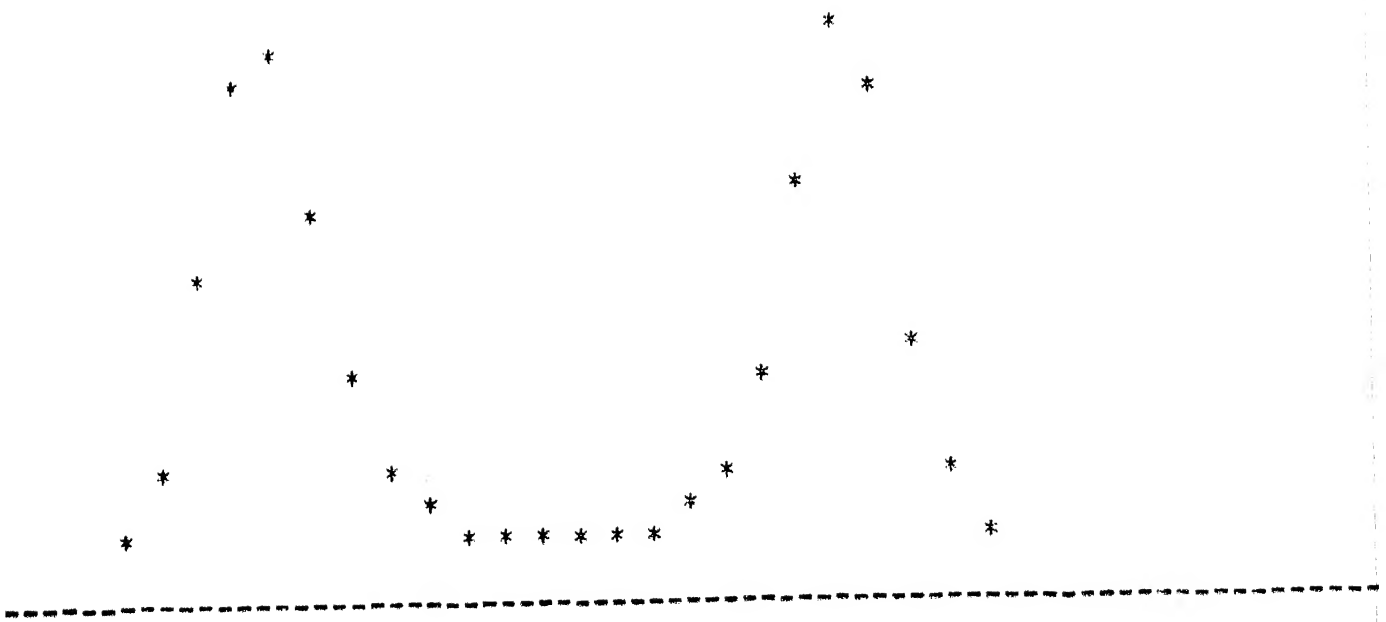
RESTORED IMAGE BY FRIEDENS MAXIMUM ENTROPY PRINCIPLE

WITH B=0.2,RO=20.0



RESTORED IMAGE BY FRIEDENS MAXIMUM ENTROPY PRINCIPLE

WITH B=0.2,RO=40.0

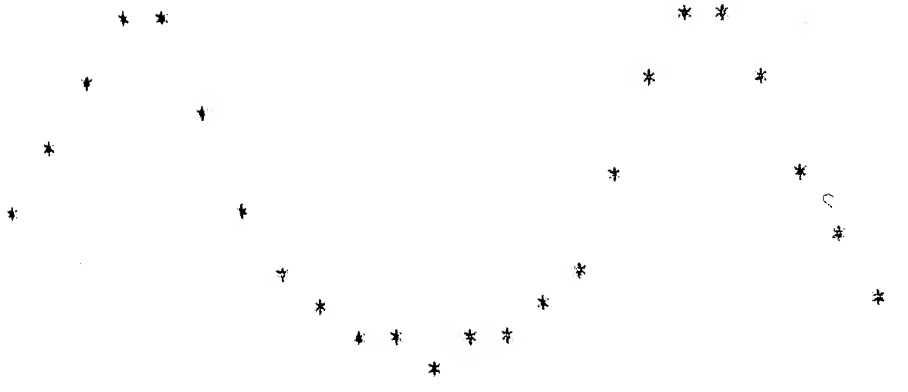


DIFFRACTION BLURRED IMAGE WITH WHITE GAUSSIAN NOISE N(0,0.5)

S/N=23 DB

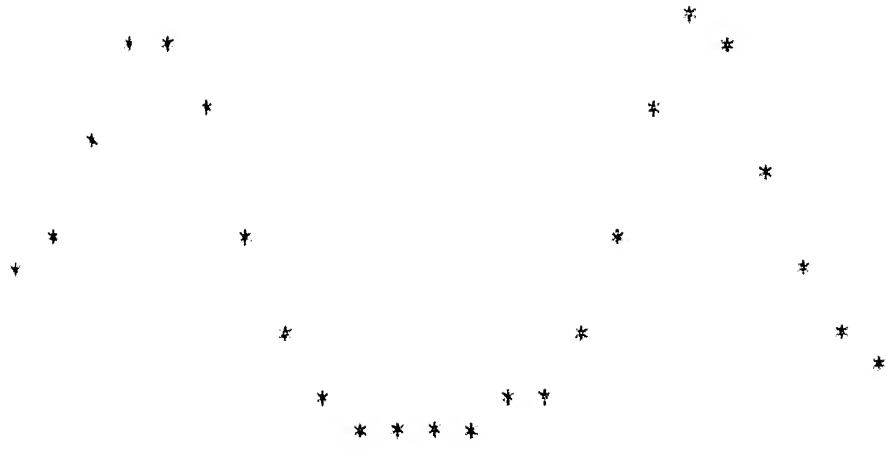
RESTORED IMAGE BY FRIEDENS MAXIMUM ENTROPY PRINCIPLE

WITH B=1.0,K0=10.0



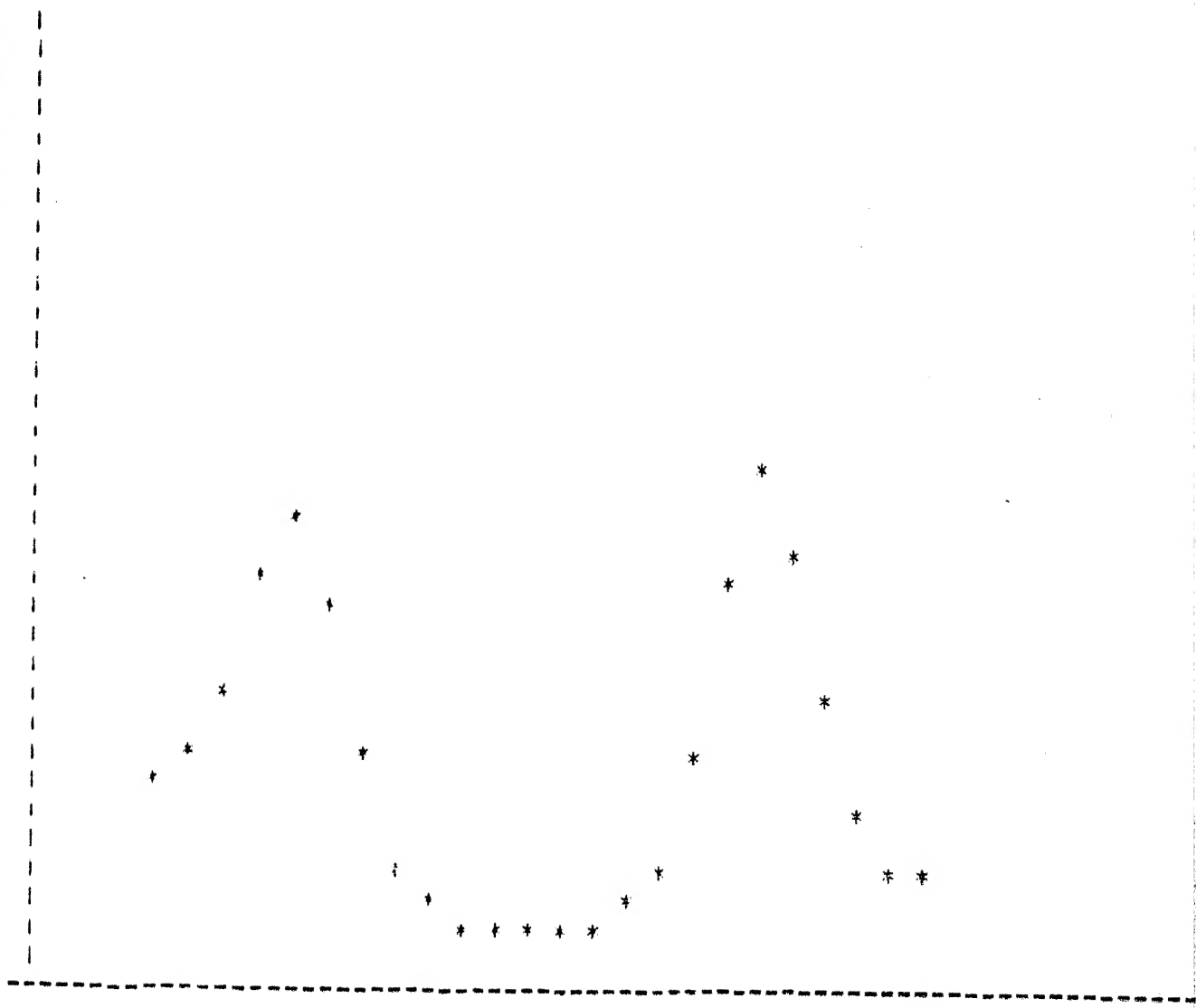
RESTORED IMAGE BY FRIEDENS MAXIMUM ENTROPY PRINCIPLE

WITH B=1.0,AM=20.0



RESTORED IMAGE BY BURG'S MAXIMUM ENTROPY PRINCIPLE

SNR = 1.0, M=40.0



```
C     ------------------------------------------------------------
C     PROGRAM TO FILTER THE DEGRADED IMAGE MATRIX FROM THE PICTURE MATRIX
C
C     BY PASSING IT THROUGH TWO DIMENSIONAL DIFFRACTION BLUR FILTER
C     WITH SEPARABLE IMPULSE RESPONSE ((SIN(X)**2)*(SIN(Y)**2)).
C     THE FILTER COEFFICIENTS ARE NORMALIZED SUCH THAT THEIR SUM IS 1.
C     ------------------------------------------------------------
      INTEGER PIC(9,9)
      REAL HT,HINR,HR(5),HC(5),DR(12,9),DC(12,9),
     1 SUMA,DMT(9,12),X(12,9),XIMG(12,12),IMG(12,12)
      OPEN (UNIT=21,DEVICE='DSK',FILE='PIC.DAT')
      M=9
      N=5
      K=M+N-1
      READ(21,*) ((PIC(I,J),J=1,9),I=1,9)
      PI=3.14159
      PI2=3.14159
      R=(N-1)/2+1
      DO 10 I=1,N
      IF(I.EQ.I)GOTO 7
      ALPHA =(I-R)+T*PI/R/K
      HR(I)=(SIN (ALPHA)/ALPHA)**2
      GOTO 8
7     HR(I)=1.0
8     HC(I)=HR(I)
10    CONTINUE
      SUM=0.0
      DO 11 I=1,N
      DO 12 J=1,N
      SUM=SUM+HR(J)*HC(I)
12    CONTINUE
11    CONTINUE
      SUM=SQRT(SUM)
      DO 13 I=1,N
      HR(I)=HR(I)/SUM
      HC(I)=HR(I)
13    CONTINUE
      DO 20 I=1,M
      DO 20 J=1,M
      DR(I,J)=0.0
      DC (I,J)=0.0
20    CONTINUE
      DO 40  I=1,M
      DO 50  I=1,N
      DR(I+J-1,J)=HR(I)
      DC(I+J-1,J)=HC(I+J-1,J)
50    CONTINUE
40    CONTINUE
      WRITE(15,*),((DR(I,J),J=1,M),I=1,M)
      DO 60  I=1,M
      DO 70  I=1,M
```

```
      DPT(J,I)=DP(I,J)
70    CONTINUE
60    CONTINUE
      DO 80 I=1,M
      DO 90 J=1,N
      SUM=0.0
      DO 100 K=1,N
      SUM=SUM+DC(I,K)*PIC(K,J)
100   CONTINUE
      X(I,J)=SUM
90    CONTINUE
80    CONTINUE
      AMAX=0.0
      DO 120 I=1,M
      DO 125 J=1,M
      SUM=0.0
      DO 130 K=1,N
      SUM=SUM+X(I,K)*DPT(K,J)
130   CONTINUE
      TMG(I,J)=SUM
      AMAX=AMAX+SUM
125   CONTINUE
120   CONTINUE
      WRITE(30,*) ((TMG(I,J),J=1,M),I=1,M)
      STOP
      END
```

```
C     ------------------------------------------------------------
C     FRIEDEN'S MAXIMUM ENTROPY RESTORATION FOR 2-D IMAGES.
C     ------------------------------------------------------------
C     PROGRAM MER2D.FOR
      DIMENSION DR(12,8),SUM(64),S(144,64),IMG(144),F(145)
      DIMENSION DEBAF(145,145),LAMDA(145),SROW(64),XIMG(12,12)
      REAL S,F,DEBAF,LAMDA,RO,SIGMA,MAX,SROW,IO,DR,SUM,S,IMG
     1 ,XIMG
      INTEGER NOOFI,MX,NX,M1,M2,M3,SINT(144,3),YIMG(8,8)
      COMMON DR,SROW,F,DEBAF,LAMDA,SUM,MAX,IMG,S,RO,S,SINT,IO
      SIGMA=0.2
      M=8
      N=12
      MX=M**2
      NX=N**2
      R=2*SIGMA
      NOOFI=10
      IO=0.0
      RO=20.0
      OPEN (UNIT=21,DEVICE='DSK',FILE='FOR31.DAT')
      READ (21,*),((XIMG(I,J),J=1,12),I=1,12)
C     COLUMN STACKING OF DEGRADED IMAGE
      K=1
      DO 5 J=1,12
      DO 5 I=1,12
      IMG(K)=XIMG(I,J)
      IO=IO+XIMG(I,J)
      K=K+1
5     CONTINUE
      OPEN(UNIT=22,DEVICE='DSK',FILE='FOR35.DAT')
      READ(22,*),((DR(I,J),J=1,8),I=1,12)
C     INITIALIZATIONS OF LAMDAS
      DO 10 I=1,MX
      LAMDA(I)=0.0
10    CONTINUE
      LAMDA(MX+1)=-1-ALOG(IO/NX)
C     INITIALISATION OF ESTIMATED DATA
      DO 11 I=1,MX
      SUM(I)=EXP(-1-LAMDA(MX+1))
11    CONTINUE
C     FORMULATION OF THE CONVOLUTION MATRIX S
      DO 20 I=1,MX
      KT=I
      CALL ROW(KT,M1,M2,M3)
      DO 30 I=1,64
      S(I,J)=SROW(J)
30    CONTINUE
      SINT(I,1)=M1
      SINT(I,2)=M2
      SINT(I,3)=M3
20    CONTINUE
      CALL GEIF
```

```
      DO 40 I=1,NOOFI
      CALL GFUNCR
      CALL GETLAM
      CALL GR.SUM
      CALL GETF
      TYPE 111,I,MAX
111   FORMAT(10 ,I6,'TH ITERATION OVER.MAXIMUM OF F IS',2X,F15.6)
40    CONTINUE
555   TOTAL=0.0
      DO 50 I=1,NX
      TOTAL=TOTAL+SUM(I)
50    CONTINUE
      RATIO=TO/TOTAL
      DO 60 I=1,8
      DO 60 J=1,8
      K=I+(J-1)+I
      YIMG(I,J)=IFIX(SUM(K)*RATIO+0.5)
60    CONTINUE
      LARGE=0
      DO 70 I=1,8
      DO 70 J=1,8
      IF(YIMG(I,J).GT.LARGE)LARGE=YIMG(I,J)
70    CONTINUE
      DO 80 I=1,8
      DO 80 J=1,8
      YIMG(I,J)=YIMG(I,J)*30/LARGE
80    CONTINUE
      WRITE (59,444),((YIMG(I,J),J=1,8),I=1,8)
444   FORMAT(8(1X,I6))
      STOP
      END


C     ------------------------------------------------------------
C     SUBROUTINE GETF COMPUTES THE RESIDUAL OF THE CONSTRAINTS
C     AFTER EACH ITERATION.
C     ------------------------------------------------------------
      SUBROUTINE GETF
      REAL S,F(145),DERAF(145,145),LAMDA(145),RO,MAX
      REAL SUM(64),S(144,64),IMG(144),DR(12,8)
      INTEGER SINT(144,3)
      COMMON DR,SROW,F,DERAF,LAMDA,SUM,MAX,IMG,S,RO,S,SINT,IO
      MAX=0.0
      DO 10 I=1,144
      F(I)=0.0
      J=SINT(I,3)
      DO 20 K=1,SINT(I,1)
      DO 30 L=1,SINT(I,2)
      F(I)=F(I)+S(I,J)*SUM(J)
      J=J+1
30    CONTINUE
      J=J+8-SINT(I,2)
20    CONTINUE
```

```
      F(I)=F(I)+EXP(-1-LAMDA(I)/RO)-B-ING(I)
      IF(MAX.LT.F(I))MAX=F(I)
10    CONTINUE
      F(145)=0.0
      DO 40 I=1,64
      F(145)=F(145)+SUM(I)
40    CONTINUE
      F(145)=F(145)-TO
      IF(MAX.LT.F(145))MAX=F(145)
      RETURN
      END


C     ------------------------------------------------------------
C     THE SUBROUTINE GETDER COMPUTES THE JACOBIAN OF F(I),
C     I=1..145, WITH RESPECT TO LAMDA(J),J=1..145.
C     ------------------------------------------------------------
      SUBROUTINE GETDER
      REAL F,IO,LAMDA,DERAF,SUM,S,MAX,RO,OR,SKOW,B,ING
      INTEGER SINT
      COMMON OR(12,8),SKOW(64),F(145),DERAF(145,145),LAMDA(145),
     1 SUM(64),MAX,ING(144),B,RO,S(144,64),SINT(144,3),TO
      DO 30 I=1,144
      DO 40 J=1,144
      DERAF(I,J)=0.0
      M=SINT(J,3)
      DO 50 K=1,SINT(J,1)
      DO 60 L=1,SINT(J,2)
      DERAF(I,J)=DERAF(I,J)-S(J,M)*S(I,M)*SUM(M)
      M=M+1
60    CONTINUE
      M=M+8-SINT(J,2)
50    CONTINUE
      IF(I.NE.J)GOTO 40
      DERAF(I,I)=DERAF(I,J)-(EXP(-1-LAMDA(I)/RO))/RO
40    CONTINUE
30    CONTINUE
      DO 80 I=1,144
      DERAF(I,145)=0.0
      DERAF(145,I)=0.0
      J=SINT(1,3)
      DO 90 K=1,SINT(I,1)
      DO 100 L=1,SINT(I,2)
      DERAF(145,I)=DERAF(145,I)-S(I,J)*SUM(J)
      DERAF(I,145)=DERAF(I,145)-S(I,J)*SUM(J)
      J=J+1
100   CONTINUE
      J=J+8-SINT(I,2)
90    CONTINUE
80    CONTINUE
      DERAF(145,145)=-F(145)-IO
      RETURN
      END
C
```

```
C       ------------------------------------------------------------
C       SUBROUTINE GETSUM COMPUTES ESTIMATED DATA AFTER EACH
C       ITERATION.
C       ------------------------------------------------------------
        SUBROUTINE GETSUM
        REAL LAMDA(145),SUM(64),S(144,64),DR(12,8),SROW(64)
        REAL F(145),DEBAF(145,145),TO,B,TMG(144)
        INTEGER SINT(144,3)
        COMMON DR,SROW,F,DEBAF,LAMDA,SUM,MAX,IMG,B,KO,S,SINT,TO
        DO 10 I=1,64
        SUM(I)=0.0
        M1=(I-1)/8
        M2=I-8*M1
        M1=M1+1
        MJ=12*(M1-1)+M2
        J=MJ
        DO 20 K=1,5
        DO 30 L=1,5
        SUM(I)=SUM(I)-LAMDA(J)*S(J,I)
        J=J+1
30      CONTINUE
        J=J+12-5
20      CONTINUE
        SUM(I)=SUM(I)-1-LAMDA(145)
        SUM(I)=EXP(SUM(I))
10      CONTINUE
        RETURN
        END


C       ------------------------------------------------------------
C       SUBROUTINE GETLAM COMPUTES THE NEW VALUE OF LAMDA'S BY
C       SOLVING A SET OF 145 LINEAR EQUATIONS.
C       ------------------------------------------------------------
        SUBROUTINE GETLAM
        REAL DL(145),WKSPCE(145),S,F,DEBAF,LAMDA,SUM,MAX,TO,TMG
        REAL XF(145),B,KO
        INTEGER IA,N,IFAIL,SINT
        COMMON DR(12,8),SROW(64),F(145),DEBAF(145,145),LAMDA
     1  (145),SUM(64),MAX,TMG(144),B,KO,S(144,64),SINT(144,3),TO
        DO 10 I=1,145
        XF(I)=-F(I)
10      CONTINUE
        IA=145
        IFAIL=0
        N=145
        CALL F04ARF(DEBAF,IA,XF,N,DL,WKSPCE,IFAIL)
        IF(IFAIL.NE.0) STOP
        DO 20 I=1,145
        LAMDA(I)=LAMDA(I)+DL(I)
20      CONTINUE
        RETURN
        END
```

```
C       ------------------------------------------------------------
C       SUBROUTINE ROW FINDS THE ELEMENTS OF THE ROW,PASSED AS
C       INPUT PARAMETER,OF THE CONVOLUTION MATRIX S.IT ALSO FINDS
C       THE NON ZERO ELEMENTS IN EACH ROW TO AVOID MULTIPLICATIONS
C       WITH ZERO ELEMENTS.
C       ------------------------------------------------------------
        SUBROUTINE ROW(ROWNO,M1,M2,M3)
        INTEGER ROWNO,M1,M2,M3
        REAL DR,SKOW,F,DERAF,SUM,D,MAX,IO,IMG,LAMDA,RO
        COMMON DR(12,8),SKOW(64),F(145),DERAF(145,145),LAMDA
     1 (145),SUM(64),MAX,IMG(144),R,RO,S(144,64),START(144,3),IO
        DO 400 I=1,64
        SKOW(I)=0.0
400     CONTINUE
        K1=(ROWNO-1)/12
        K2=ROWNO-12*K1
        K1=K1+1
        IF(K1.EQ.5) GOTO 410
        IF(K1.EQ.6.OR.7.OR.8) GOTO 420
        IF(K1.LT.5) GOTO 430
        M1=5-(K1-8)
        X=K1-5
        GOTO 440
410     M1=5
        X=0
        GOTO 440
420     M1=5
        X=K1-5
        GOTO 440
430     M1=K1
        X=0
440     IF(K2.EQ.5)GOTO 450
        IF(K2.EQ.6.OR.7.OR.8)GOTO 455
        IF(K2.LT.5)GOTO 460
        M2=5-(K2-8)
        M3=8*X+K2-5+1
        Y=K2-5
        GOTO 470
455     M2=5
        M3=8*X+K2-5+1
        Y=K2-5
        GOTO 470
450     M2=5
        M3=8*X+1
        Y=0
        GOTO 470
460     M2=K2
        M3=8*X+1
        Y=0
470     J=M3
```

```
      L1=X+1
      L2=L1+M1-1
      L3=I+1
      L4=L3+M2-1
      DO 500 K=L1,L2
      DO 510 L=L3,L4
      SKO(J)=DK(K1,K)*DR(K2,L)
      J=J+1
510   CONTINUE
      J=J+M-M2
500   CONTINUE
      RETURN
      END
C     ------------------------------------------------------------------
C     ------------------------------------------------------------------
```

```
C     ------------------------------------------------------------
C     THIS PROGRAM PLOTR.FOR  PLOTS THE TWO DIMENSIONAL PICTURES.
C     ------------------------------------------------------------
      INTEGER*2 IA
      DIMENSION IA(64,64),ID(4096)
      OPEN(UNIT=21,DEVICE='DSK',FILE='FOR32.DAT')
      READ(21,*),((IA(I,J),J=1,64),I=1,64)
      NX=64;NY=64;LAW=1;IL=0;IH=31;NEG=1;LG=42
      CALL DSP(IA,NX,NY,LAW,IL,IH,NEG,LG)
      RETURN
      END
      SUBROUTINE DSP(IA,NX,NY,LAW,IL,IH,NEG,LG)
C     COMMON IA(64,64)
      INTEGER*2 IA
      INTEGER*2 IB(64,64),LEV(32),BLANK(5)
      LOGICAL*1 LINE(128,5),GRAY(32,5)
      DIMENSION IA(64,64)
      DATA GRAY/6*'M',5*'W','X','M','X','U',
     1'Z','W','M','N','O','S','=','T','*','+',
     2'+','=',':','-','.','-',' ',6*'N',3*'*',
     3'*','+','+',4*'-',5*' ','=',' ',' ','-',
     443*' ','-',3*' ',4*':','O','O','+','-',
     5524*' ','O','O','O',29*' ','+',31*' '/
      GN=32.0
      FL=IL
      FH=IH
      IF((FH-FL).GT.0.0) GO TO 100
      T=FL
      FL=FH
      FH=T
100   RANGE=(FH-FL+1)/GN
      AA=(SQRT(FH)-SQRT(FL))/GN
      EE=(FH-FL)/ALOG(GN+1.0)
      T=AMAX1(FL,1.0)
      SS=-(1.0/GN)*ALOG(FL/T)
      DO 160 I=1,32
      GO TO (110,120,130,140),LAW
110   FLEV=FL+(I-1)*RANGE+0.5
      GO TO 150
120   FLEV=(SQRT(FL)+(I-1)*AA)**2+0.5
      GO TO 150
130   FLEV=FL+EE*ALOG(FLOAT(I))+0.5
      GO TO 150
140   FLEV=FH+EXP(SS*(GN-I))+0.5
150   LEV(I)=FLEV
160   CONTINUE
      IF(NX.GT.64) NX=64
      IF(NY.GT.64) NY=64
      DO 180 I=1,NX
      DO 180 J=1,NY
      KLT=1
      DO 170 K=1,32
      IF (IA(I,J).GE.LEV(K)) KLT=K
170   CONTINUE
```

```
      IB(I,J)=KOUT
160   CONTINUE
      WRITE(LG,1)
      IX=NX
      IY=2*NY
      DO 210 I=1,IX
      DO 190 J=1,5
      BLANK(J)=0
190   CONTINUE
      DO 200 K=2,IY,2
      J=K/2
      NG=IB(I,J)
      IF(NEG.EQ.0) NG=33-NG
      DO 200 L=1,5
      LINE(K-1,L)=GRAY(NG,L)
      LINE(K,L)=GRAY(NG,L)
      IF(NG.NE.32) BLANK(L)=1
200   CONTINUE
      WRITE(LG,2)
      DO 210 L =1,5
      IF(BLANK(L).EQ.0) GO TO 210
      WRITE (LG,3) (LINE(N,L),N=1,IY)
210   CONTINUE
1     FORMAT(1H1)
2     FORMAT(1H )
3     FORMAT(1H+,3X,129A1)
      RETURN
      END
```

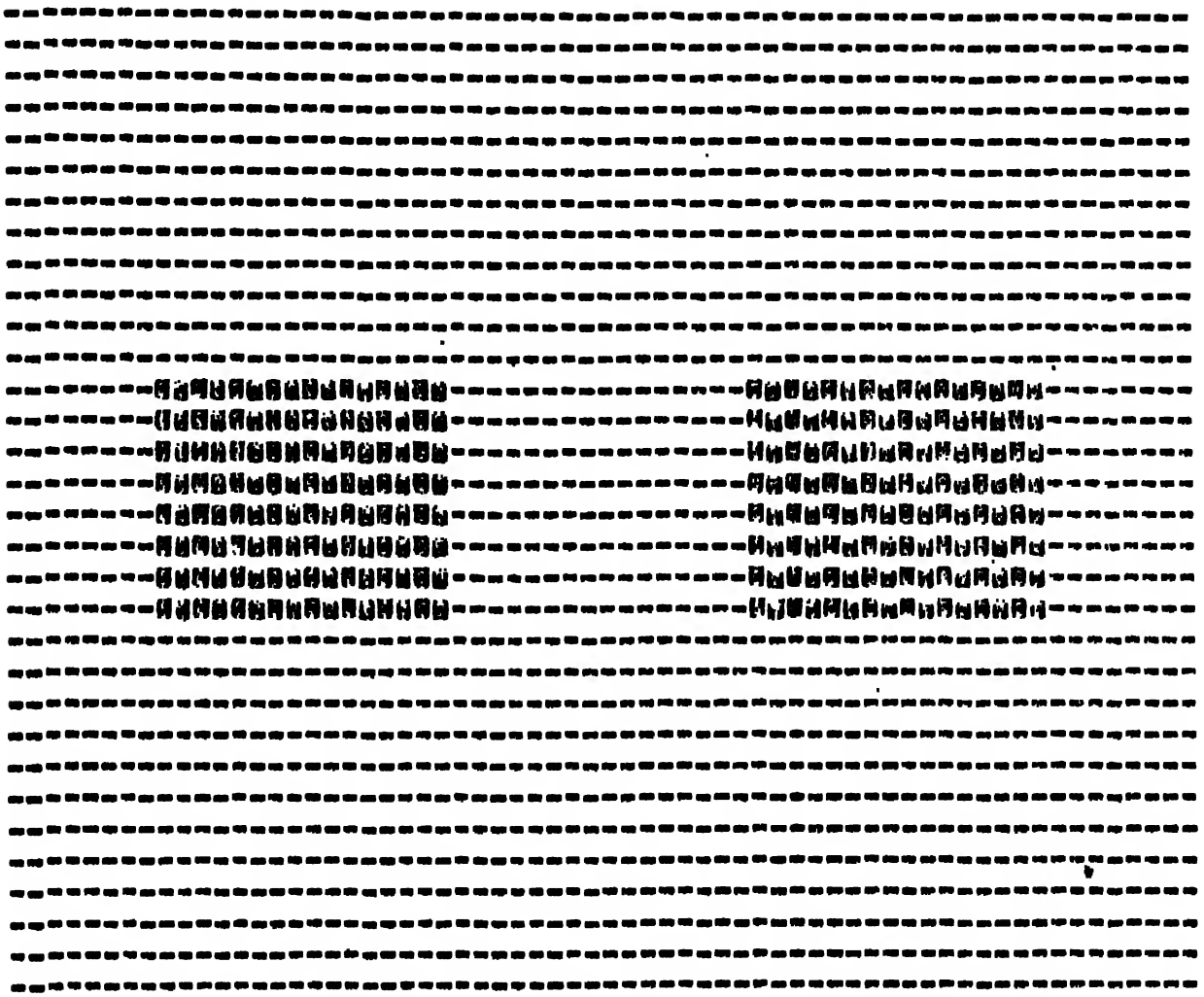
OBJECT IMAGE MATRIX OF SIZE 8*8

| | | | | | | | |
|---|---|---|---|---|---|---|---|
| 30 | 30 | 30 | 30 | 30 | 30 | 30 | 30 |
| 30 | 30 | 30 | 30 | 30 | 30 | 30 | 30 |
| 30 | 30 | 30 | 30 | 30 | 30 | 30 | 30 |
| 30 | 5 | 5 | 30 | 30 | 5 | 5 | 30 |
| 30 | 5 | 5 | 30 | 30 | 5 | 5 | 30 |
| 30 | 30 | 30 | 30 | 30 | 30 | 30 | 30 |
| 30 | 30 | 30 | 30 | 30 | 30 | 30 | 30 |
| 30 | 30 | 30 | 30 | 30 | 30 | 30 | 30 |

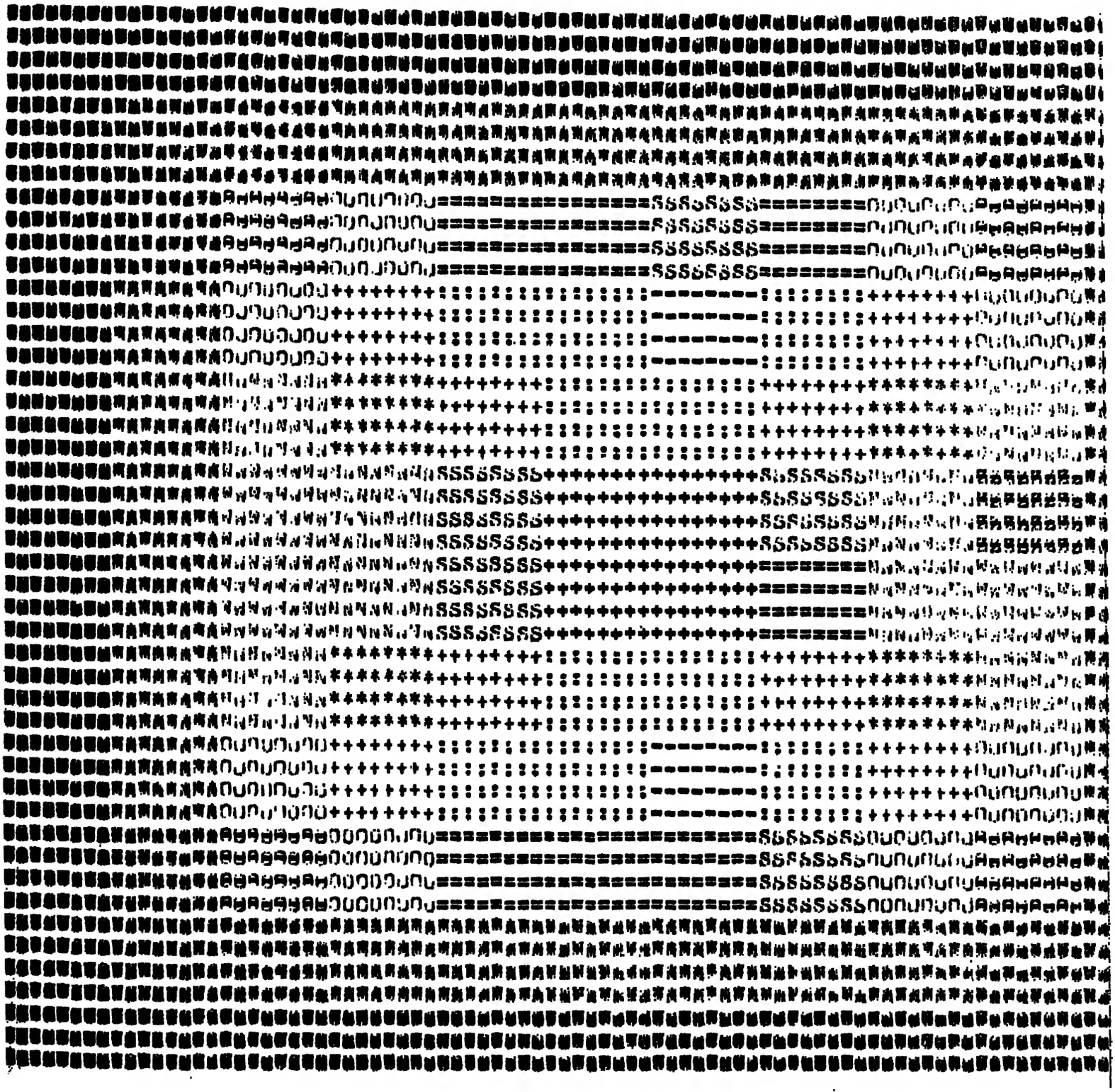
RESTORED IMAGE MATRIX WITH K0=20

| | | | | | | | |
|---|---|---|---|---|---|---|---|
| 17 | 21 | 19 | 20 | 20 | 18 | 22 | 16 |
| 22 | 29 | 23 | 23 | 24 | 23 | 30 | 23 |
| 20 | 18 | 13 | 22 | 24 | 14 | 16 | 21 |
| 19 | 7 | 6 | 20 | 20 | 6 | 6 | 19 |
| 19 | 7 | 6 | 20 | 19 | 6 | 6 | 19 |
| 21 | 16 | 15 | 23 | 22 | 15 | 17 | 20 |
| 22 | 26 | 23 | 23 | 24 | 24 | 27 | 22 |
| 17 | 21 | 19 | 21 | 20 | 19 | 21 | 19 |

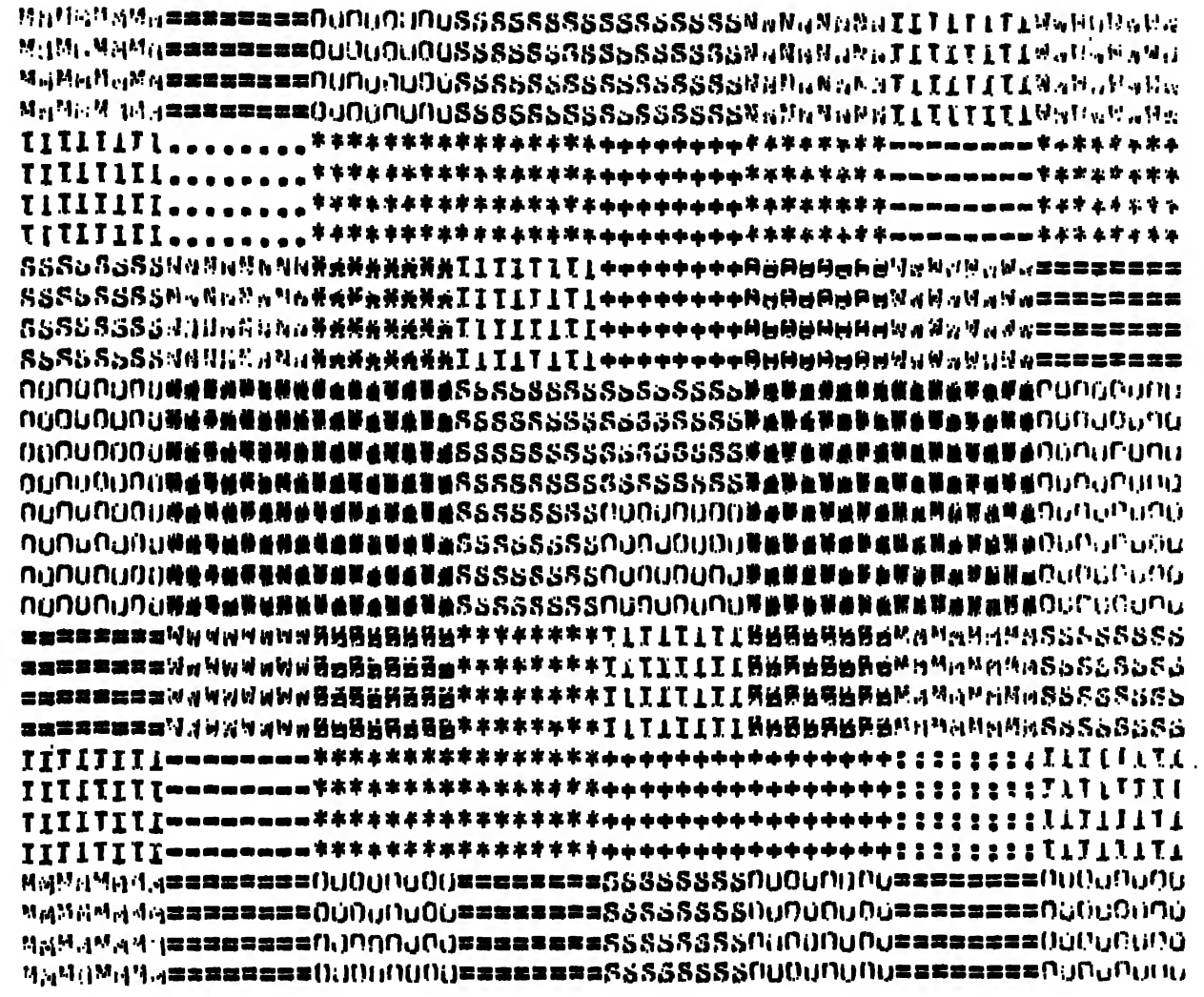
OBJECT IMAGE

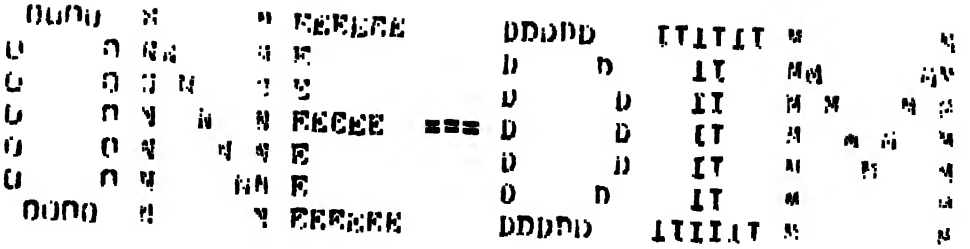


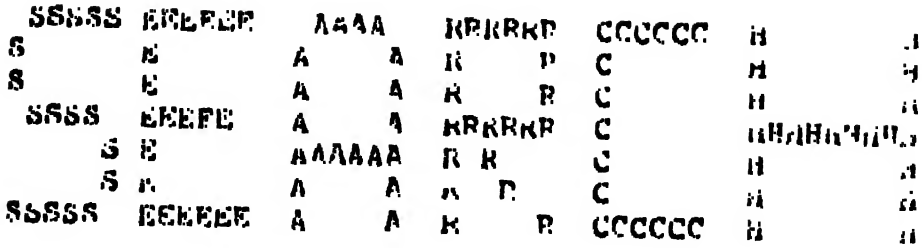
DEGRADED IMAGE WITH NOISE

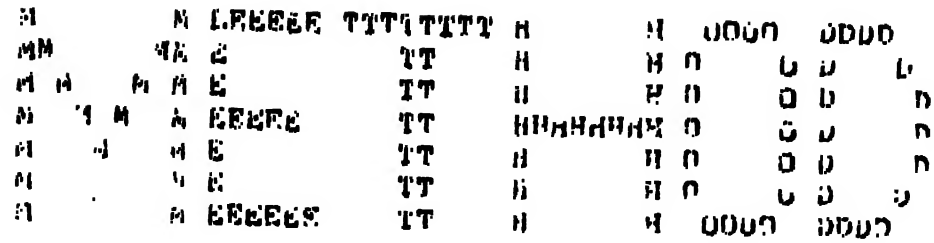


RESTURED IMAGE WITH RU=20




OOOO X X EEEEEE
O O XX X E
O O X X X E
O O X X X EEEEE
O O X X E
O O X XX E
OOOO X X EEEEEE
===
DDDDD IIIIII M M
D D II MM MM
D D II M M M M
D D II M M M
D D II M M
D D II M M
DDDDD IIIIII M M


SSSSS EEEEEE AAAA RRRRR CCCCCC H H
S E A A R R C H H
S E A A R R C H H
SSSS EEEEE A A RRRRR C HHHHH
S E AAAAAA R R C H H
S E A A R R C H H
SSSSS EEEEEE A A R R CCCCCC H H

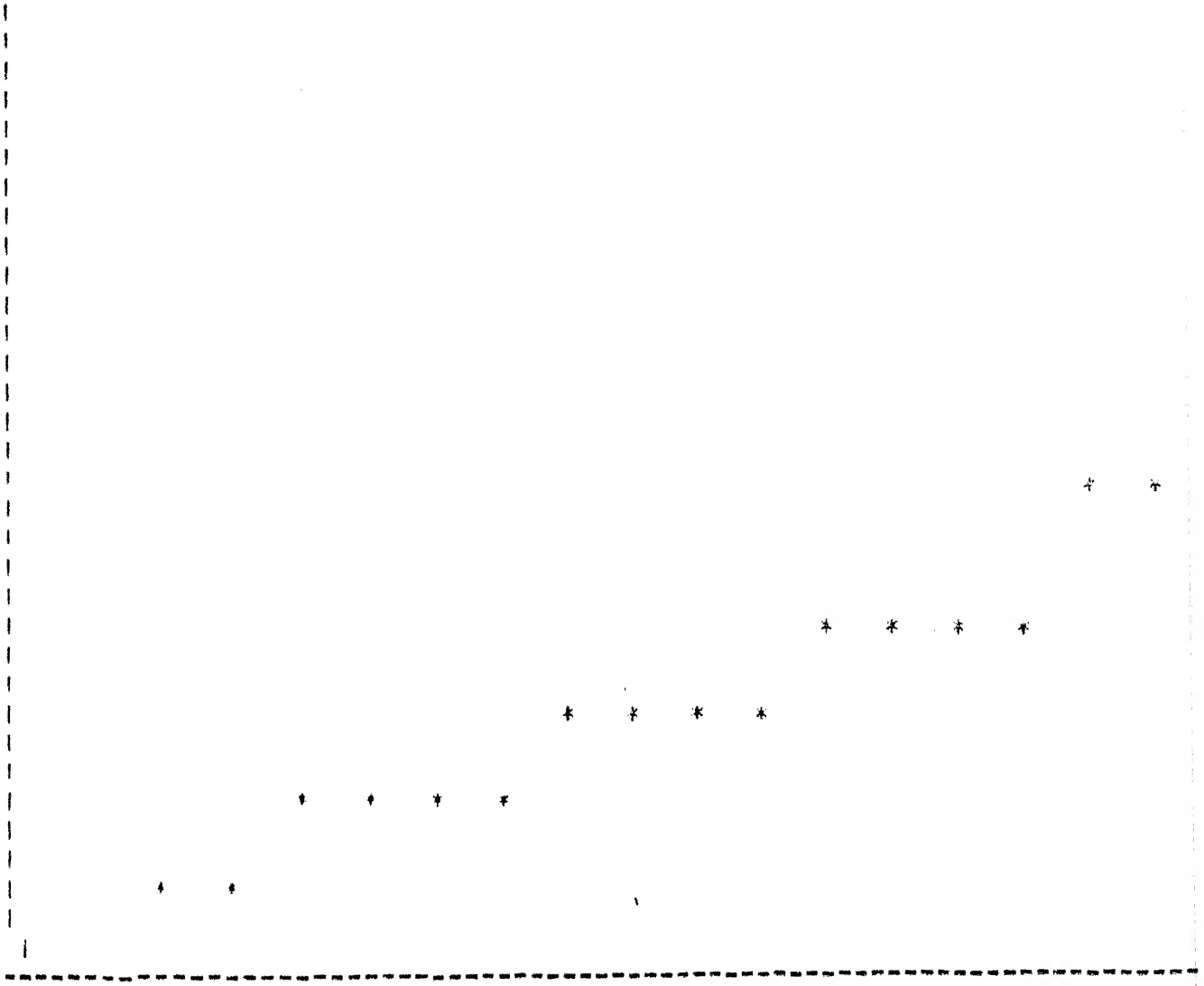

N LEEEEE TTTTTTT H H OOOO OOOO
NN NN E TT H H O O O O
N N N E TT H H O O O O
N N N EEEEE TT HHHHHHH O O O O
N N E TT H H O O O O
N N E TT H H O O O O
N EEEEEE TT H H OOOO OOOO

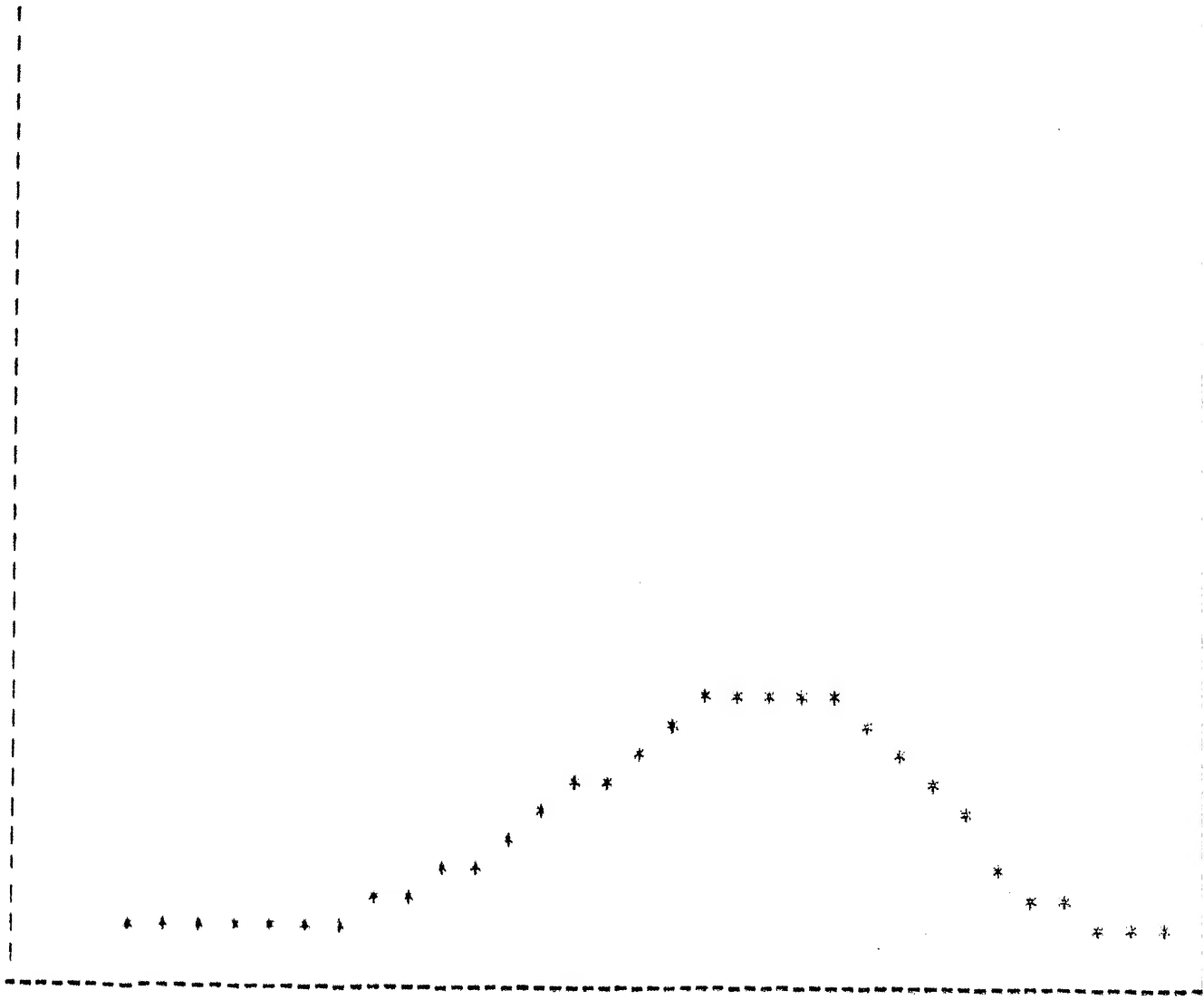
```
C     ------------------------------------------------------------------
C     MAXIMUM ENTROPY RESTORATION BY SOLVING A SYSTEM OF INITIAL VALUE
C     DIFFERENTIAL EQUATIONS.THIS PROGRAM MER.FOR SEARCHES ALONG A REAL
C     DEFINED PATH,AND HENCE IS AN ONE DIMENSINAL SEARCH PROBLEM.THE
C     INITIAL VALUE OF MU,WHICH BELONGS TO AN OPEN INFINITE INTERVAL ,
C     IS FOUND BY TRIAL FOR QUICKER CONVERGENCE.FOR THIS REASON,THE
C     VALUE OF MU AND THE NO.OF ITERATIONS IS FED THROUGH AN ACCEPT
C     STATEMENT. THE INPUT DATA TO THIS PROGRAM IS THE CONVOLUTION
C     MATRIX OF THE IMAGING SYSTEM AND THE NOISE INFECTED DEGRADED
C     DIFFRACTION BLURRED IMAGE DATA.
C     ------------------------------------------------------------------
C     PROGRAM MER.FOR
      DIMENSION A(64,64),D(64),F(64), FF(64),AT(64,64),ATA(64,64)
      DIMENSION X(64),XX(64),AMAT(50,2)
      REAL LAMO,LAMN,MU,INCR,EPS,QF,SIGMA,SUM,AA,B,C,T
      INTEGER N,M,NOIT,PIC(64),L
      COMMON/ARRAYS/A,D,F,FF,ATA,XX
      COMMON/VAR/QF,LAMO,LAMN
      COMMON/CONST/N,M,L,SIGMA
      N=16
      L=17
      M=N+L-1
      SIGMA=0.25
      EPS=0.01
      OPEN(UNIT=23,DEVICE='DSK',FILE='FOR34.DAT')
      OPEN(UNIT=25,DEVICE='DSK',FILE='FOR36.DAT')
      READ(23,*) ((A(I,J),J=1,N),I=1,M)
C     A(M,N) IS THE CONVOLUTION MATRIX OF THE IMAGING SYSTEM.
      READ(25,*) (D(I),I=1,M)
C     D(M) IS THE DEGRADED IMAGE WITH NOISE N(0.0,SIGMA).
      T=0.0
      DO 5 I=1,M
      T=T+D(I)
5     CONTINUE
      AA=0.0
      B=0.0
      C=0.0
      DO 10 J=1,M
      SUM=0.0
      DO 20 I=1,N
      SUM=SUM+A(J,I)
20    CONTINUE
      B=B+SUM*D(J)
      SUM=SUM**2
      AA=AA+SUM
      C=C+D(J)**2
10    CONTINUE
      AA=0.5*AA/SIGMA
      B=-B/SIGMA
      C=0.5*C/SIGMA
      DISC=ABS(B)**2-4*AA*(C-0.5*M)
      TYPE *,DISC,AA,B,C
      IF(DISC.LT.0.0) GOTO 44
      ALPHA1=(-B-SQRT(DISC))/(2*AA)
```

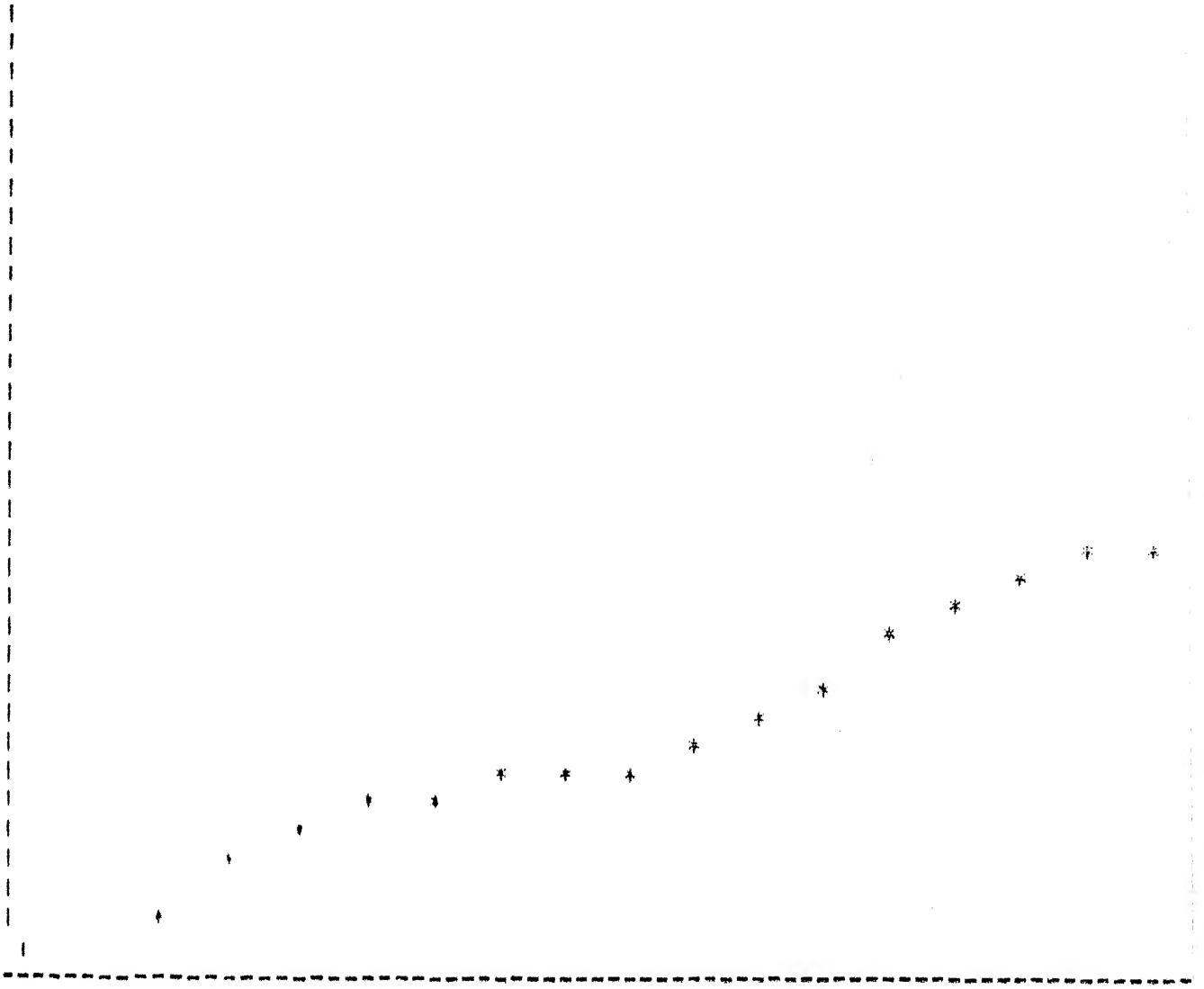
```
      ALPHA2=(-B+SQRT(DISC))/(2*AA)
      MU=1+ALOG(ALPHA2)+1.0
      GOTO 33
44    ACCEPT*,MU
33    CONTINUE
      DO 25 I=1,N
      XX(I)=0.0
      DO 26 J=1,M
      AT(I,J)=A(J,I)
      XX(I)=XX(I)+AT(I,J)*D(J)
26    CONTINUE
      XX(I)=XX(I)/SIGMA
25    CONTINUE
      DO 27 I=1,N
      X(I)=0.0
      DO 28 J=1,N
      SUM=0.0
      DO 29 K=1,M
      SUM=SUM+AT(I,K)*A(K,J)
29    CONTINUE
      ATA(I,J)=SUM/SIGMA
      X(I)=X(I)+ATA(I,J)
28    CONTINUE
27    CONTINUE
      ACCEPT*, NOIT,INCR
      DO 30 I=1,N
      F(I)=EXP(MU-1.0)
30    CONTINUE
      KK=0
      LAMO=0.0
77    CALL CALQF
      IF(ABS(QF).LE.EPS)GOTO 99
      IF(QF.GT.0.0)GOTO 31
      LAMN=LAMO-INCR
      GOTO 32
31    LAMN=LAMO+INCR
32    CONTINUE
      KK=KK+1
      IF(KK.GT.NOIT)GOTO 999
      CALL SOLVE
      DO 40 I=1,N
      F(I)=FF(I)
40    CONTINUE
      LAMO=LAMN
      GOTO 77
99    CONTINUE
      SUM=0.0
      DO 50 I=1,N
      SUM=SUM+F(I)
      PIC(I)=IFIX(F(I)+0.5)
50    CONTINUE
      TYPE*,SUM,T
      WRITE(45,*) (PIC(I),I=1,N),KK,LAMN,LAMO,SUM,T
      GOTO 55
```

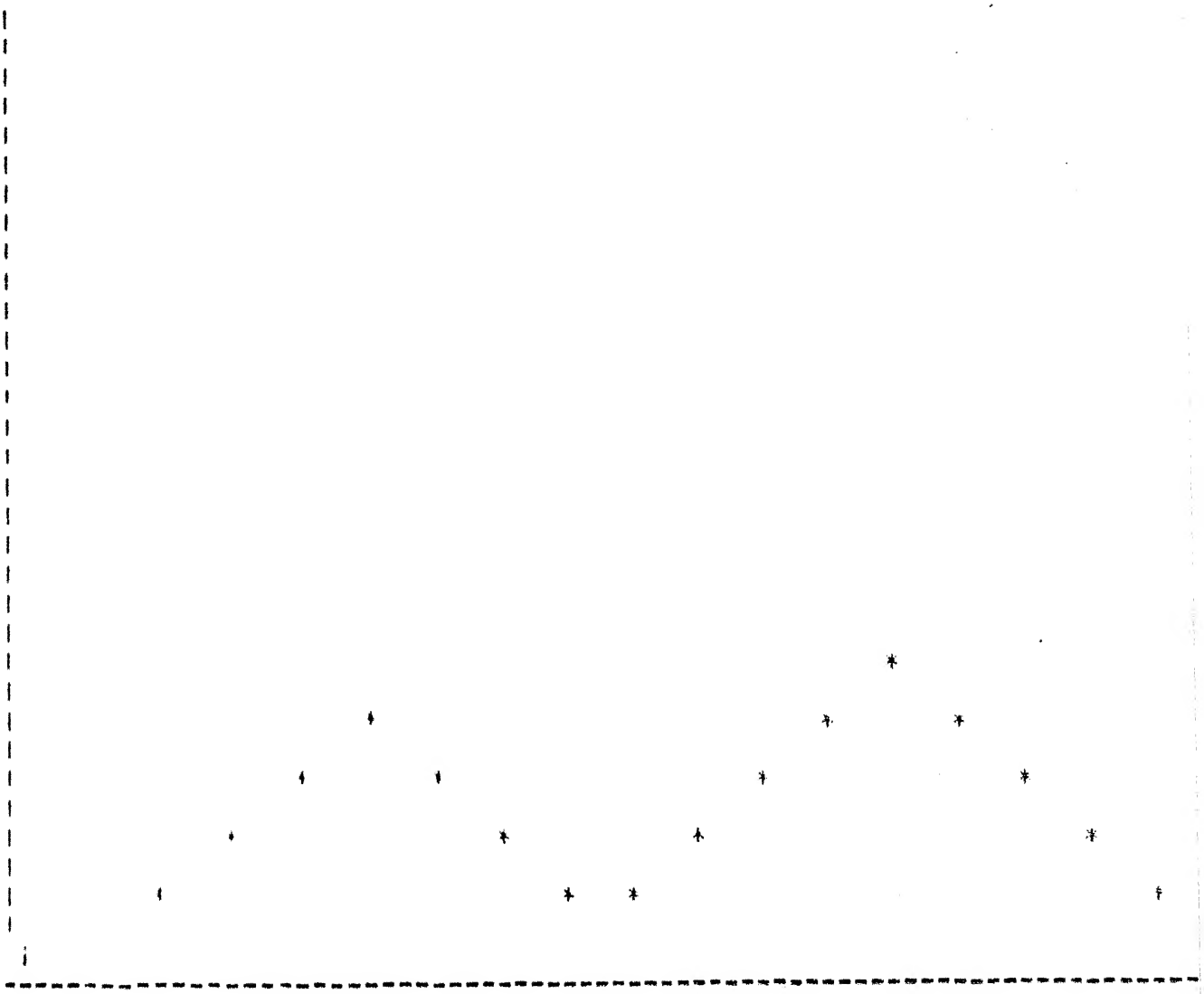
```
999     TYPE 111,KK,OF,LAMO,LAMN
111     FORMAT(1X,'NO OF ITERATIONS EXCEEDED',I6,3F15.6)
C       AFTER THE NO. OF ITERATIONS ARE EXCEEDED, FRESH VALUE OF MULT AND
C       INCREMENT INCR ARE FED DEPENDING ON THE RATE OF CONVERGENCE.
        DO 123 K=1,N
        F(I)=FF(I)
123     CONTINUE
        GOTO 77
55      STOP
        END
C       ------------------------------------------------------------
C       SUBROUTINE CALOF CALCULATES THE VALUE OF (OF=1/2M) AFTER EACH
C       ITERATION TO ACHIEVE DESIRED CONVERGENCE.
C       ------------------------------------------------------------
        SUBROUTINE CALOF
        DIMENSION A(64,64),D(64),F(64),FF(64),AF(64)
        REAL LAMO,LAMN,SIGMA,OF,SUM,XX
        INTEGER N,M,L
        COMMON/ARRAYS/A,D,F,FF,ATA(64,64),XX(64)
        COMMON/VAR/OF,LAMO,LAMN
        COMMON/CONSI/N,M,L,SIGMA
        OF=0.0
        DO 10 I=1,M
        SUM=0.0
        DO 20 K=1,N
        SUM=SUM+A(I,K)*F(K)
20      CONTINUE
        AF(I)=SUM-D(I)
        OF=OF+AF(I)*AF(I)
10      CONTINUE
        OF=0.5*((OF/SIGMA)-M)
        TYPE 555,OF
555     FORMAT(1X,'THE VALUE OF OF IS:',F15.8)
        RETURN
        END
C       ------------------------------------------------------------
C       SUBROUTINE SOLVE IS MEANT FOR SOLVING A SYSTEM OF DIFFERENTIAL
C       EQUATIONS BY GAUSS-SIEDAL ITERATIVE SCHEME.
C       ------------------------------------------------------------
        SUBROUTINE SOLVE
        DIMENSION P(64,64),BB(64),FK(64,64)
        REAL LAMO,LAMN,SIGMA,OF,A,D,F,FF,ATA,XX
        INTEGER N,M,L
        COMMON/ARRAYS/A(64,64),D(64),F(64),FF(64),ATA(64,64),XX(64)
        COMMON/VAR/OF,LAMO,LAMN
        COMMON/CONSI/N,M,L,SIGMA
        DO 10 I=1,N
        DO 10 J=1,N
        FK(I,J)=0.0
10      CONTINUE
        DO 20 I=1,N
        FK(I,I)=1.0/F(I)
20      CONTINUE
        DO 60 I=1,N
```

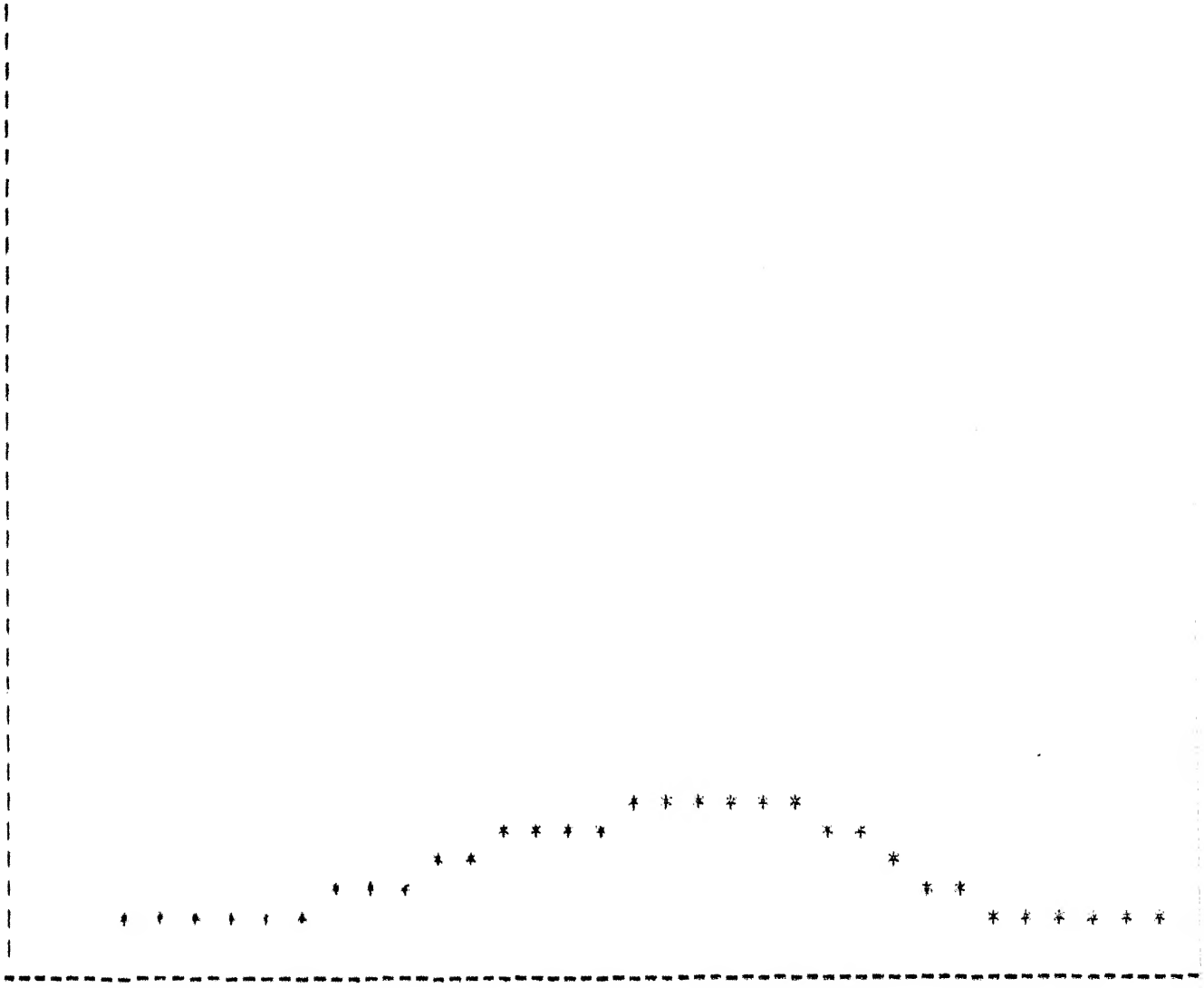
```
      DO 60 J=1,N
      P(I,J)=FK(I,J)+LAMO*ATA(I,J)
60    CONTINUE
      DO 70 I=1,N
      SUM=0.0
      DO 80 J=1,N
      SUM=SUM+ATA(I,J)*F(J)
80    CONTINUE
      BB(I)=SUM*(2*LAMO-LAMN)+1.0
70    CONTINUE
      DO 90 I=1,N
      BB(I)=BB(I)+XX(I)*(LAMN-LAMO)
90    CONTINUE
      TERM=0.0
      DO 110 J=2,N
      TERM=TERM+P(1,J)*F(J)
110   CONTINUE
      FF(1)=(BB(1)-TERM)/P(1,1)
      DO 120 I=2,N
      TERM=0.0
      DO 130 J=1,I-1
      TERM=TERM+P(I,J)*FF(J)
130   CONTINUE
      FF(I)=BB(I)-TERM
      IF(I.EQ.N)GOTO 199
      SUM=0.0
      DO 140 J=I+1,N
      SUM=SUM+P(I,J)*F(J)
140   CONTINUE
      FF(I)=FF(I)-SUM
199   FF(I)=FF(I)/P(I,I)
120   CONTINUE
      RETURN
      END
C     ------------------------------------------------------------------
C     ------------------------------------------------------------------
```

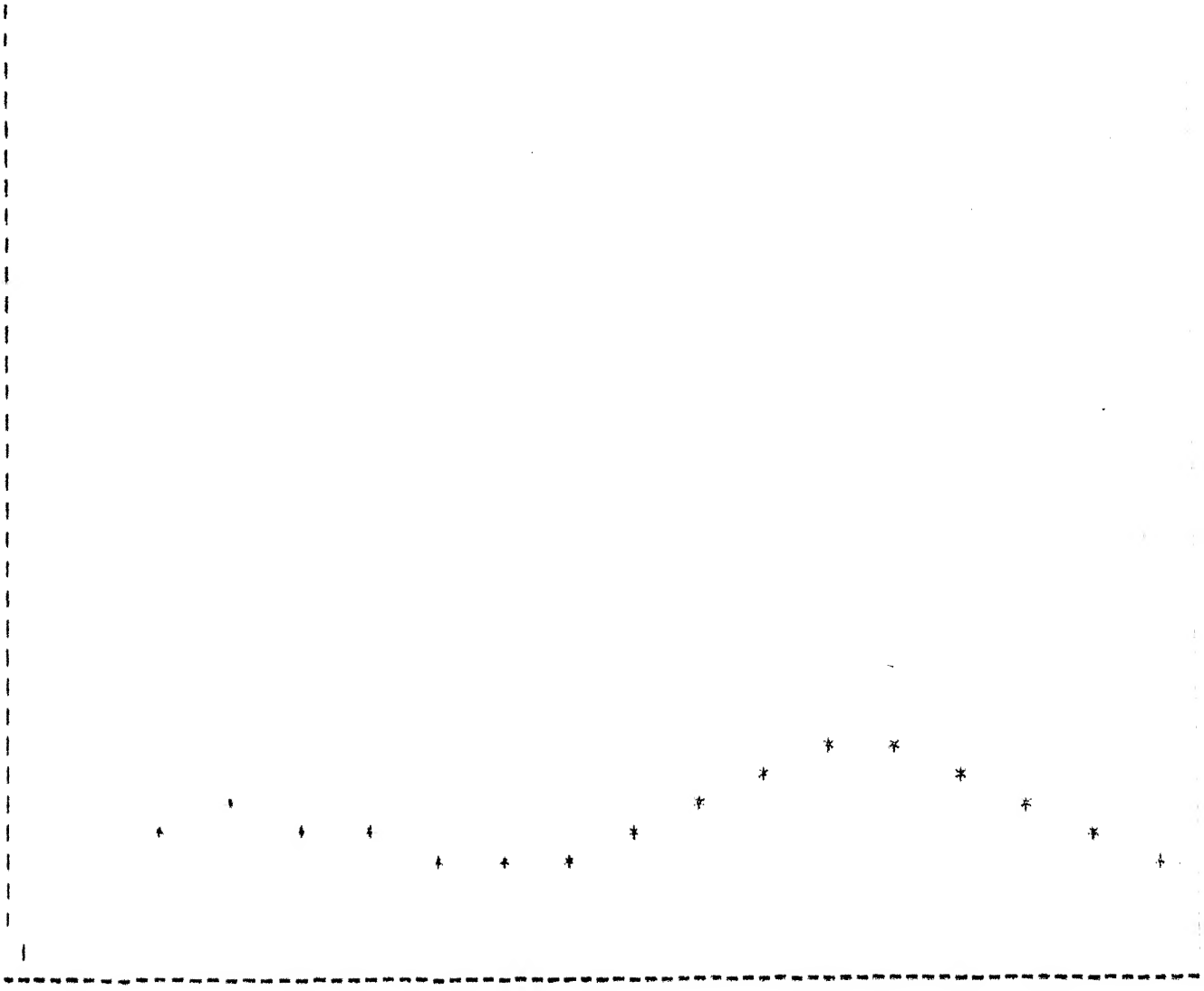




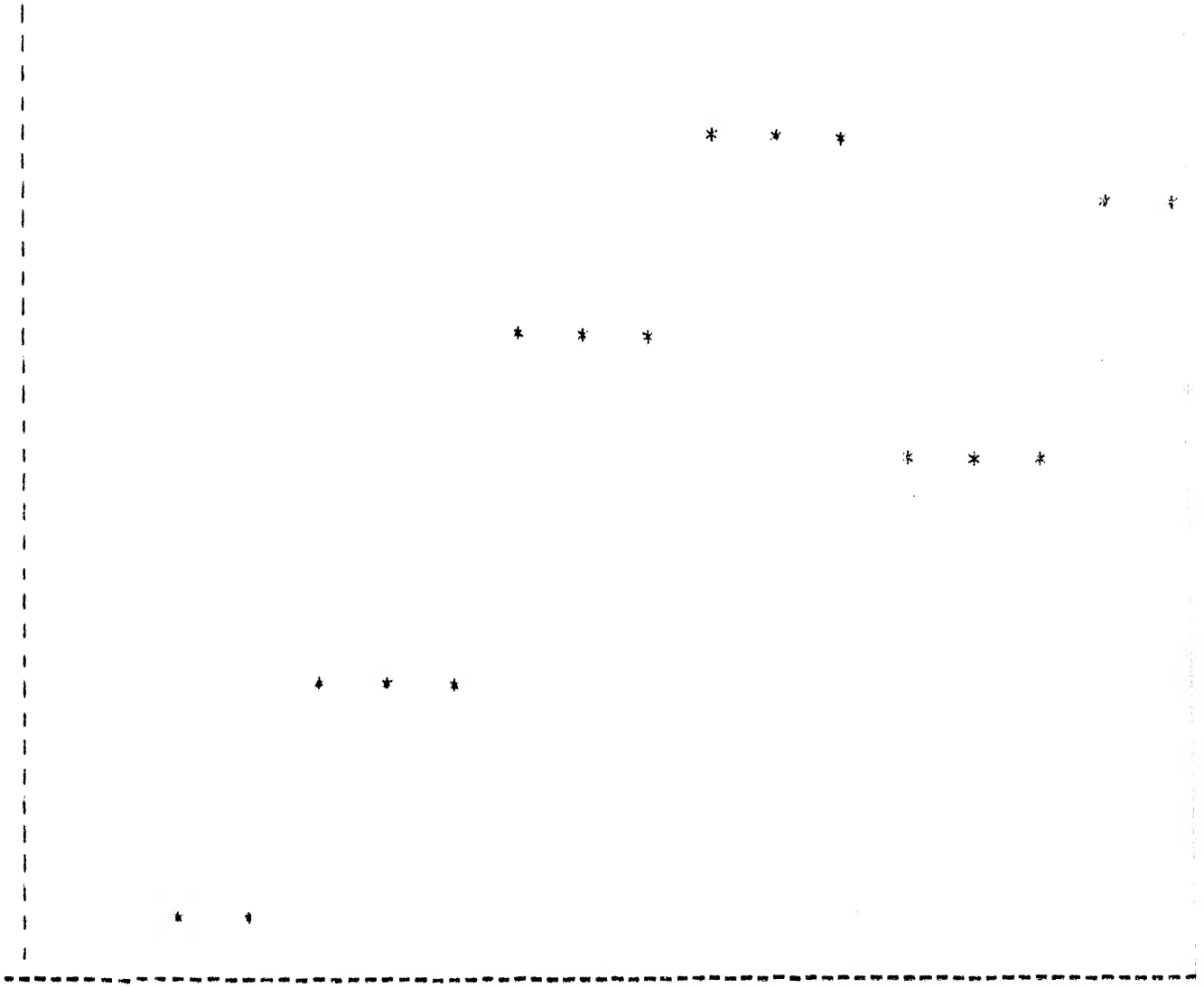


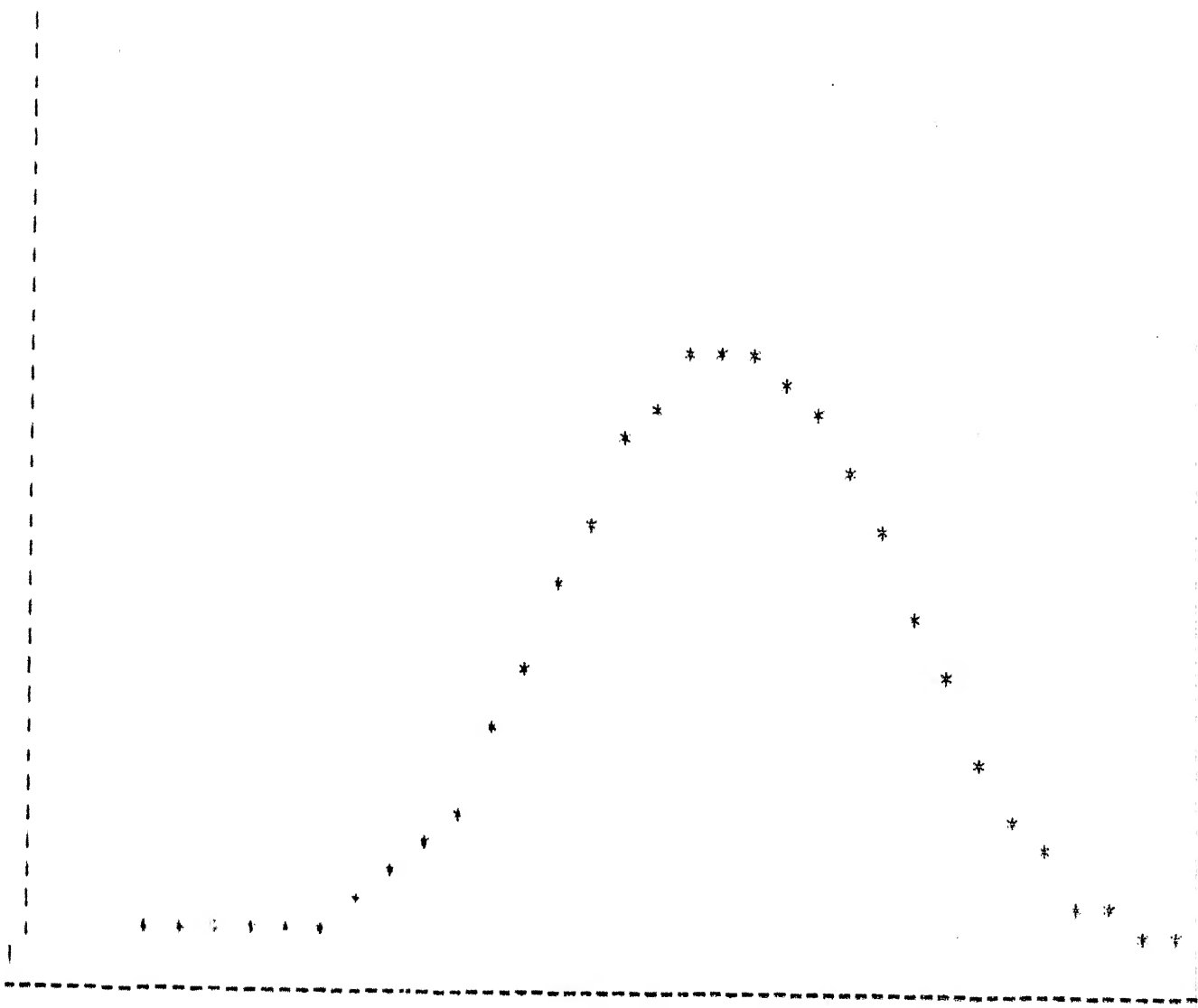


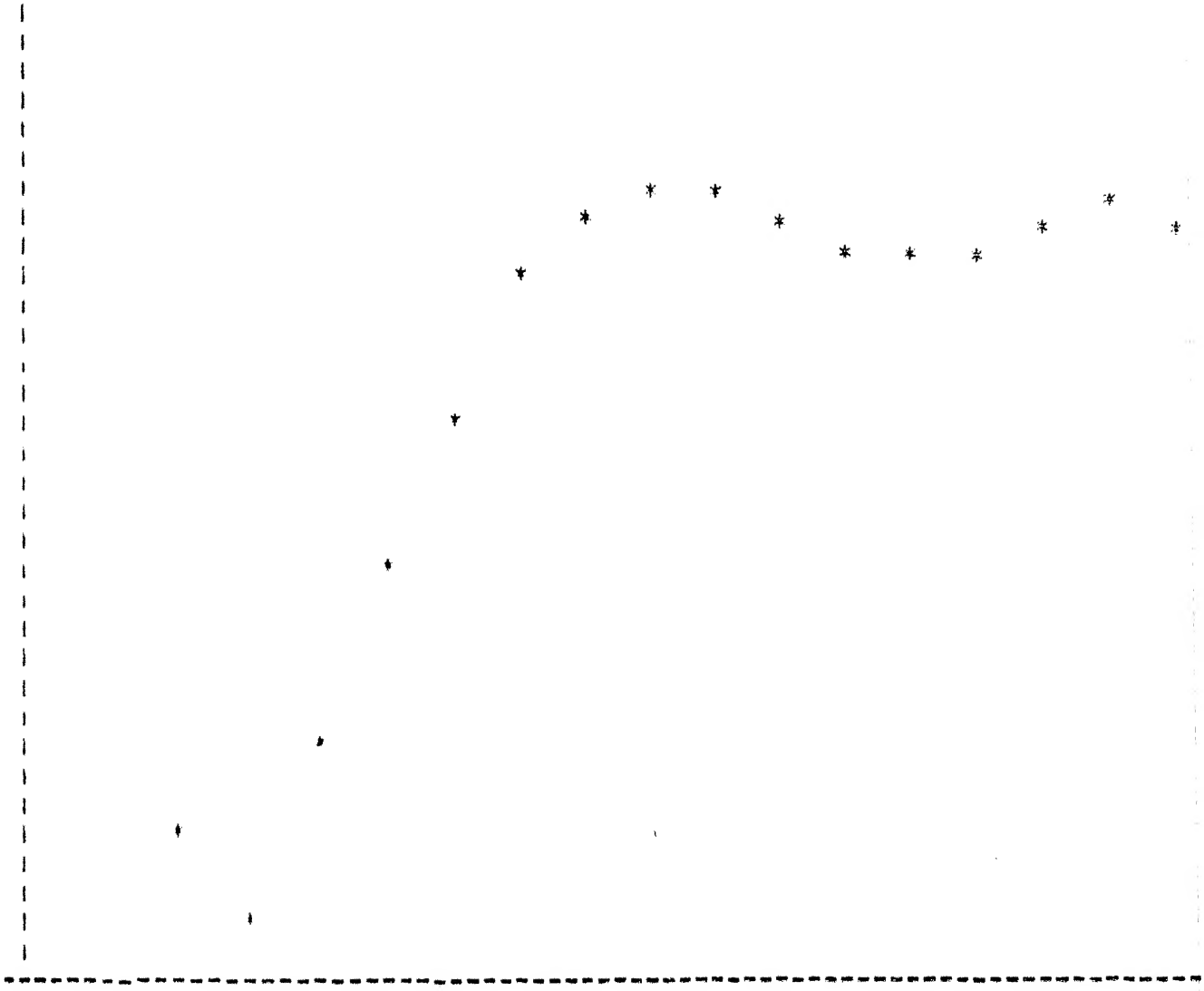




THE BEST PAGE OF SIZE 16

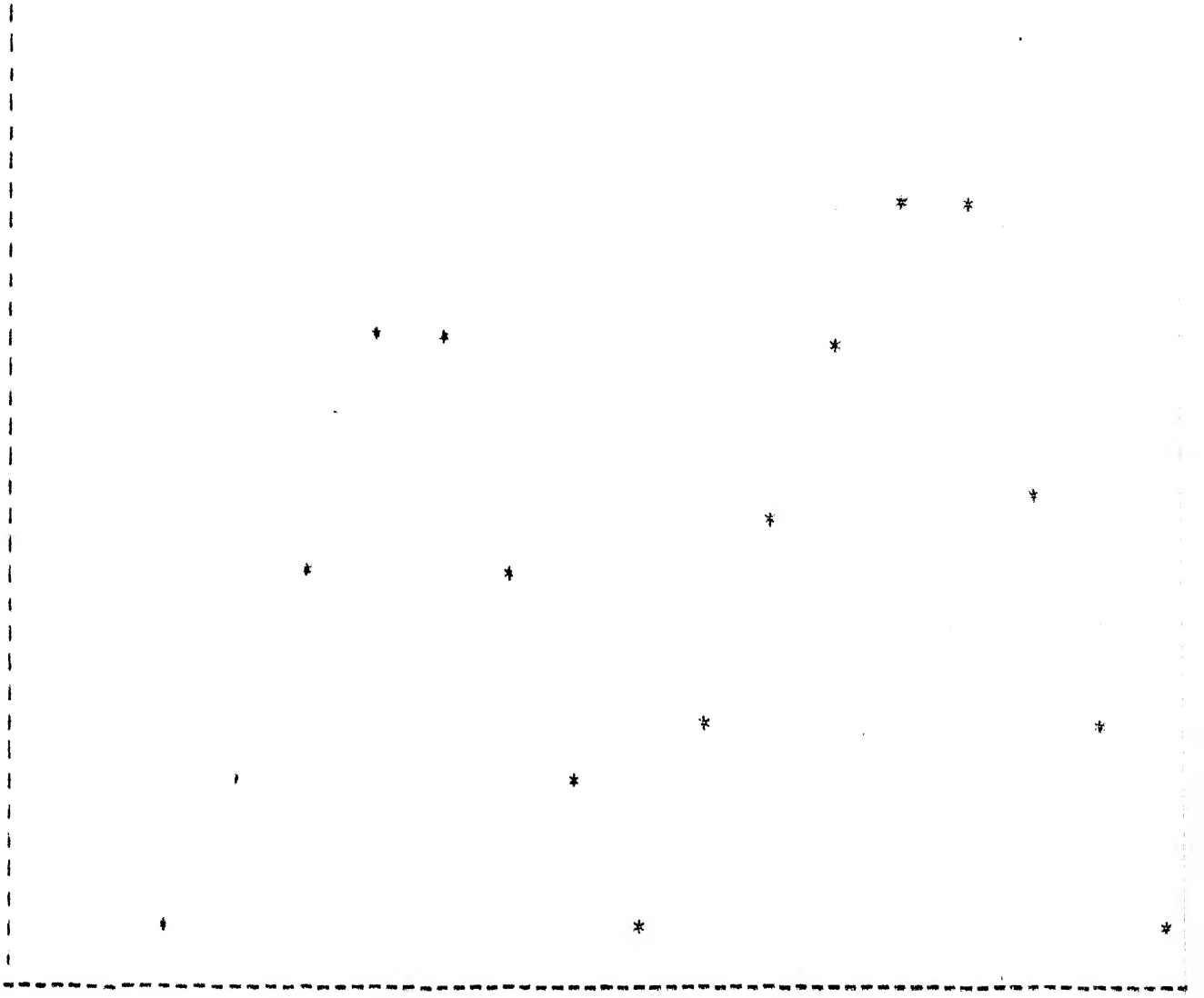






[illegible] FRIEDEN'S ALGORITHM

[illegible]=0.1, P0=70



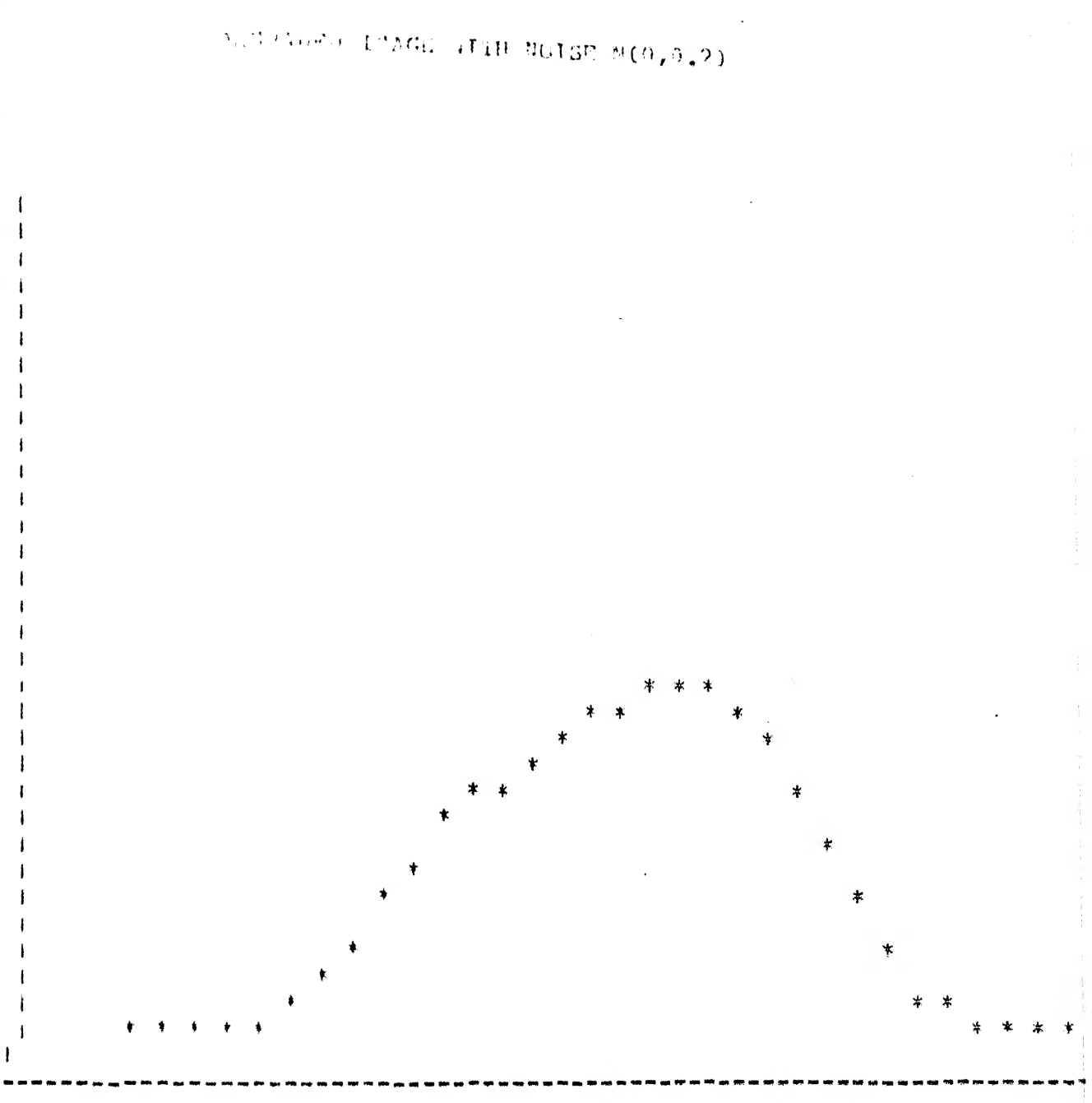
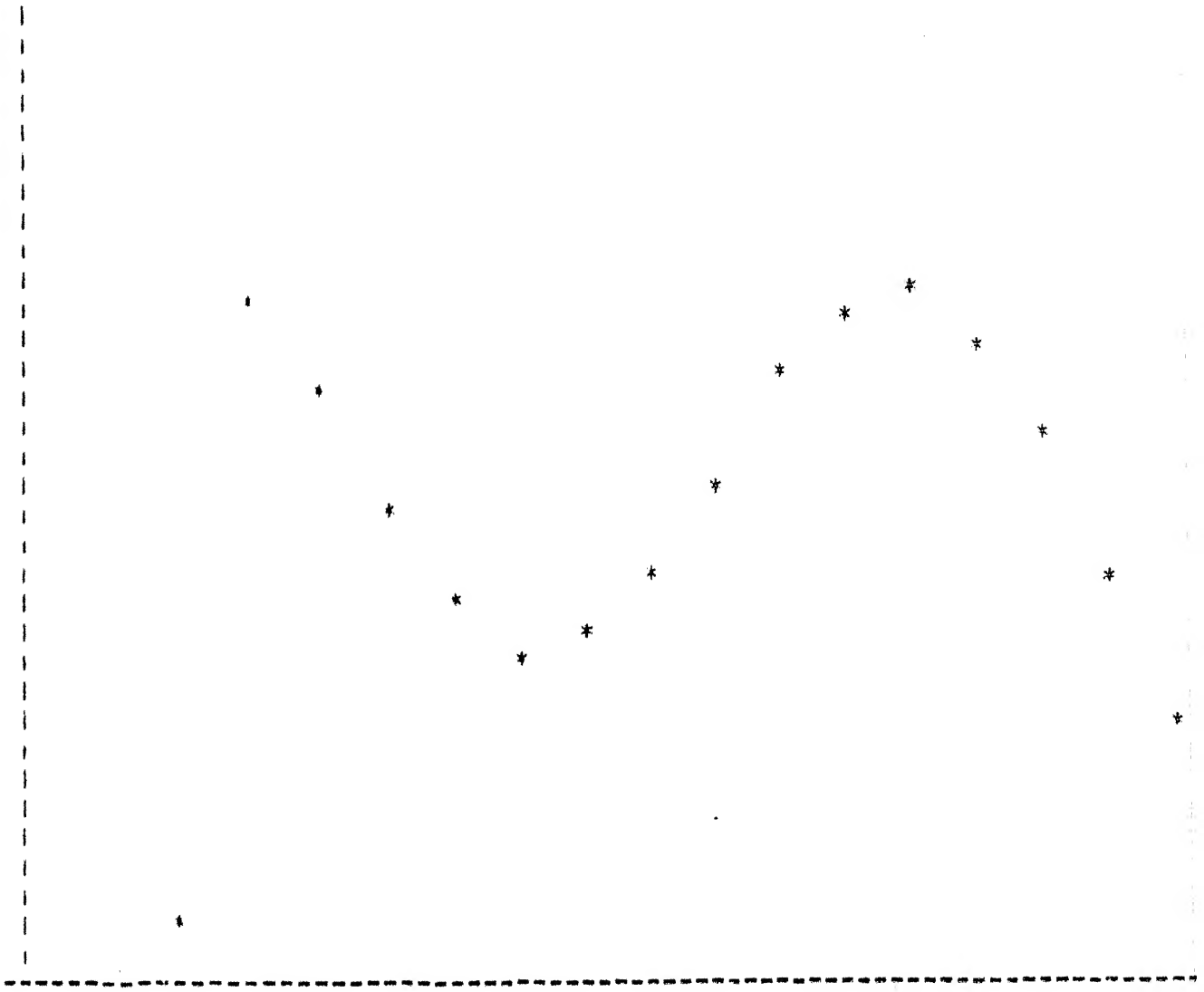
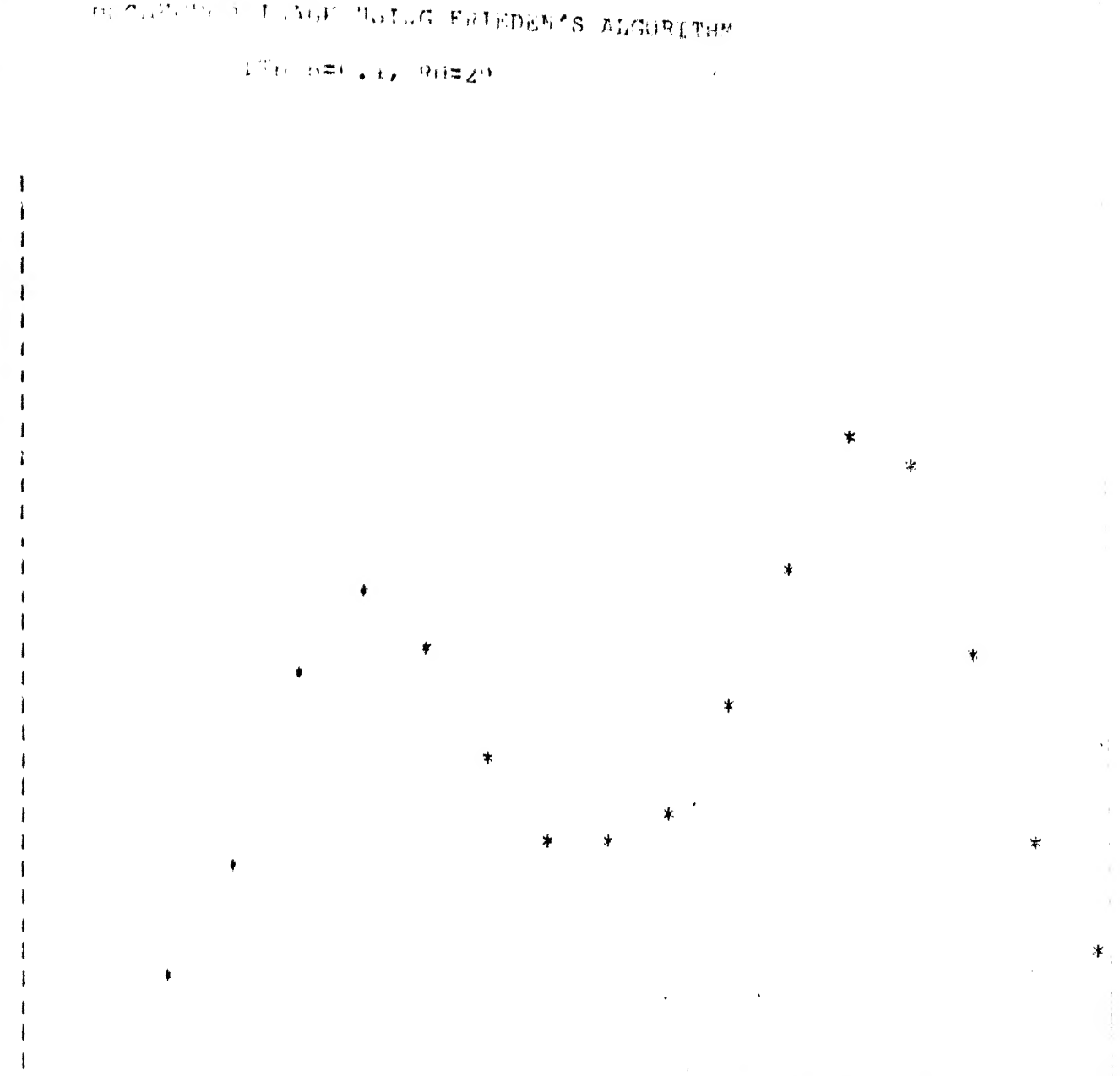
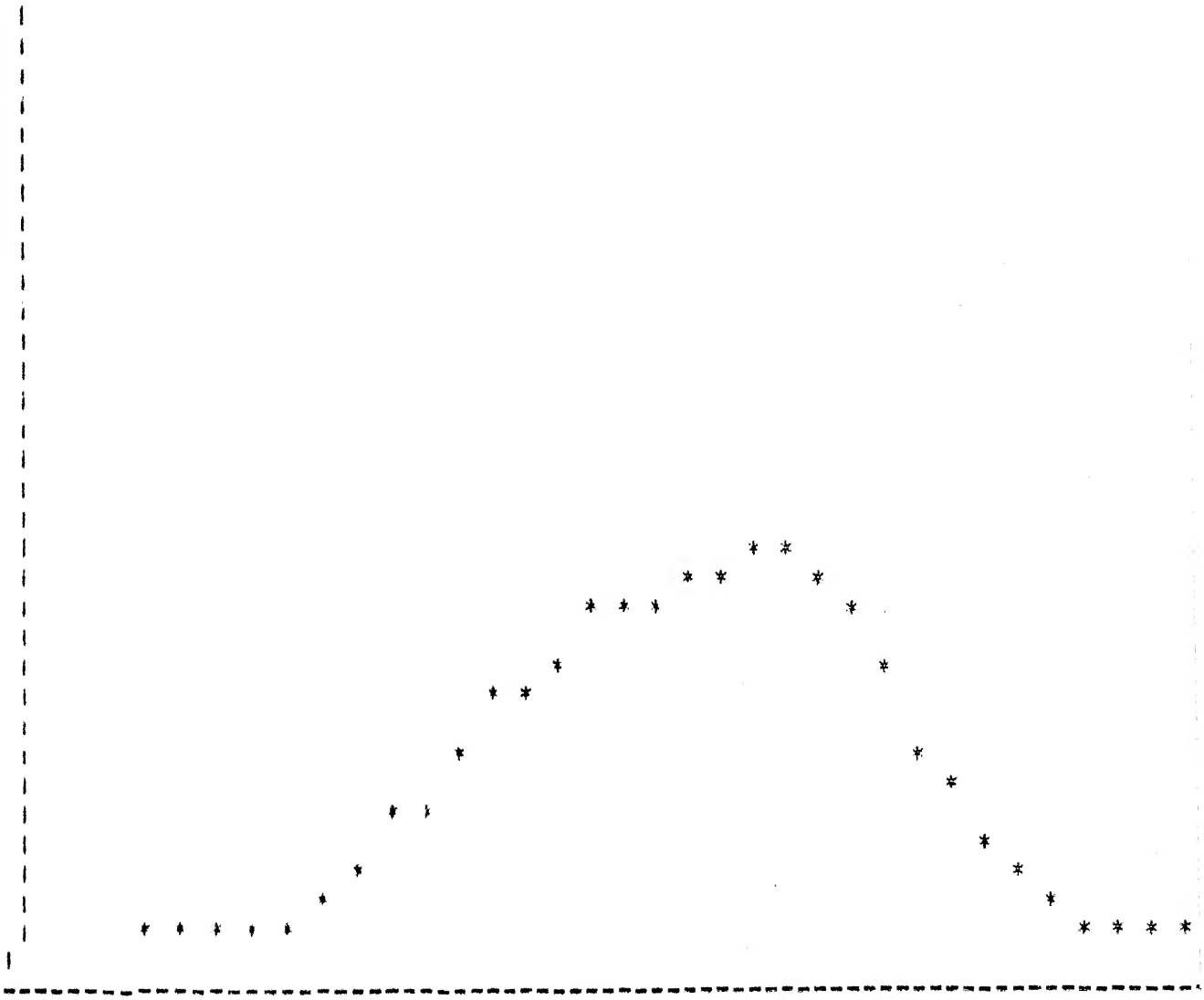


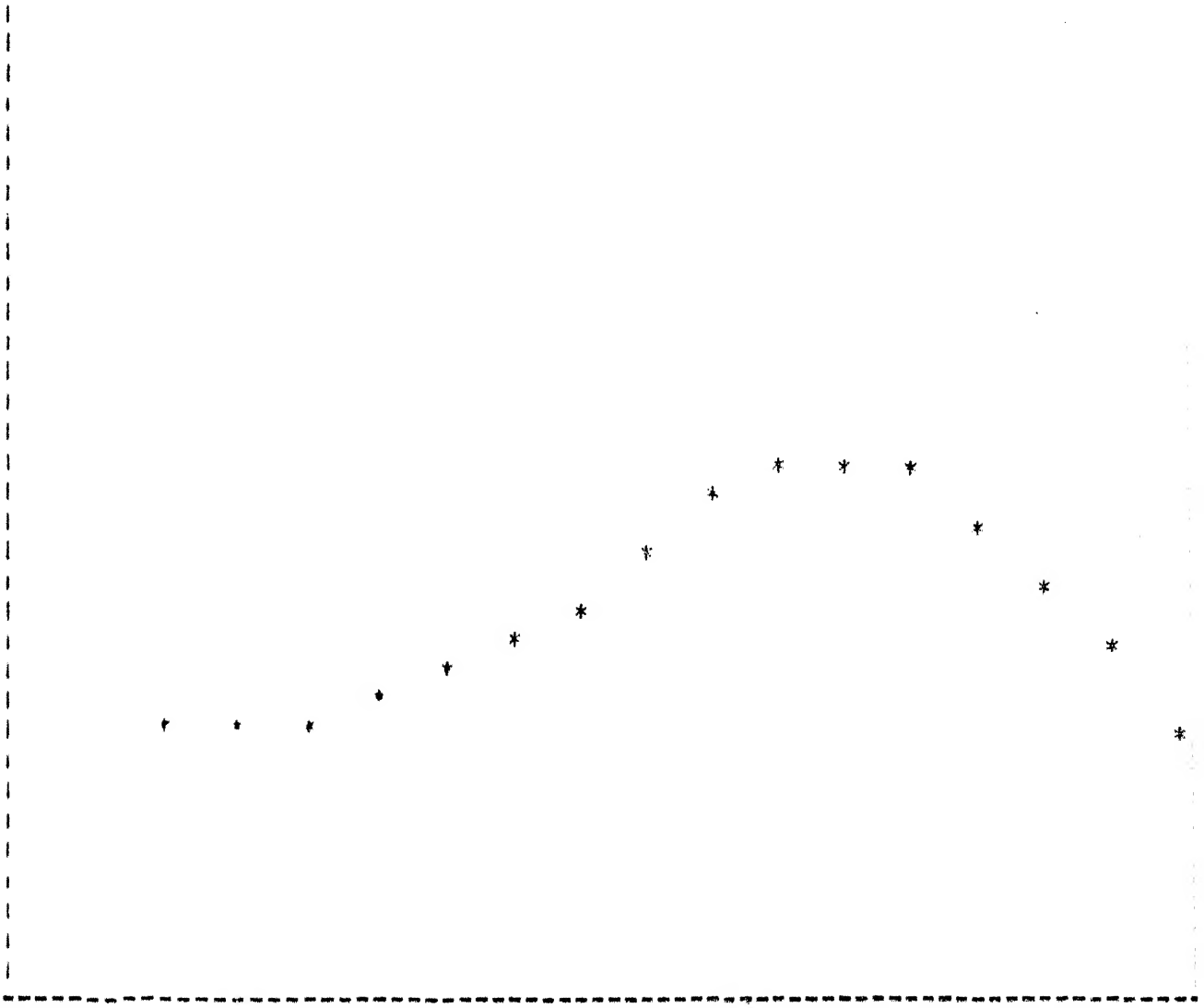
IMAGE WITH NOISE N(0,0.2)

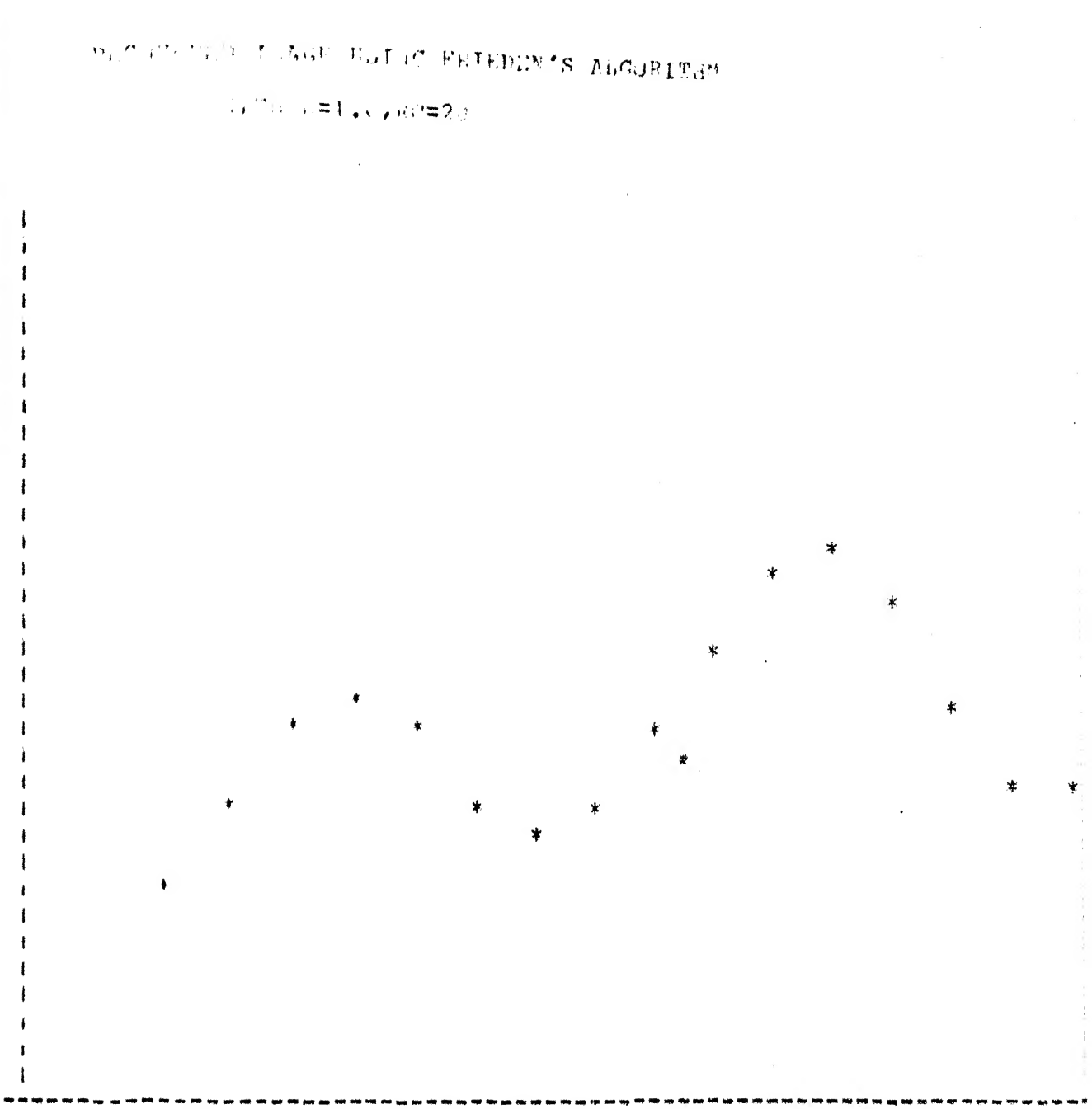




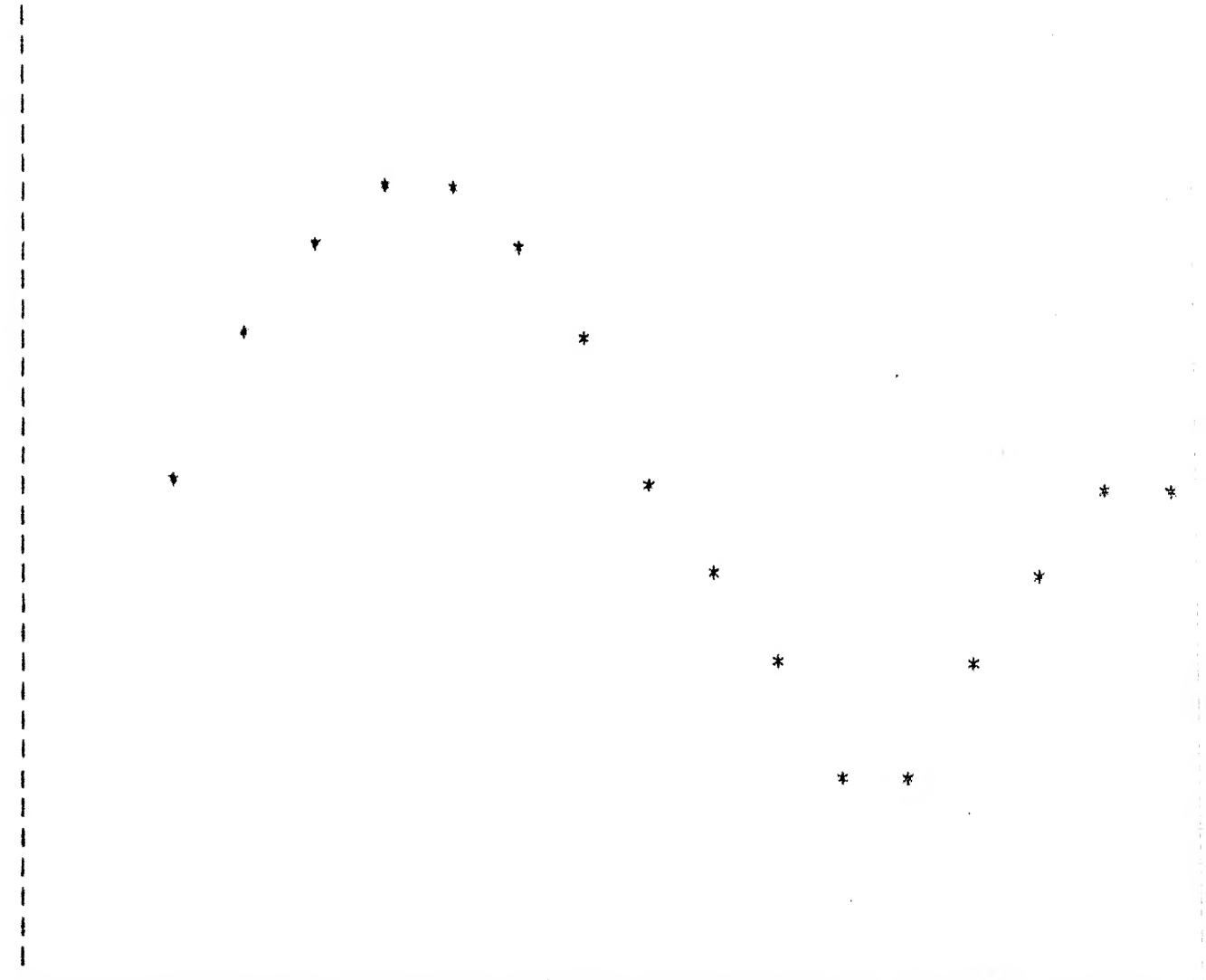
[illegible] AVEC BRUIT N(0,0.5)



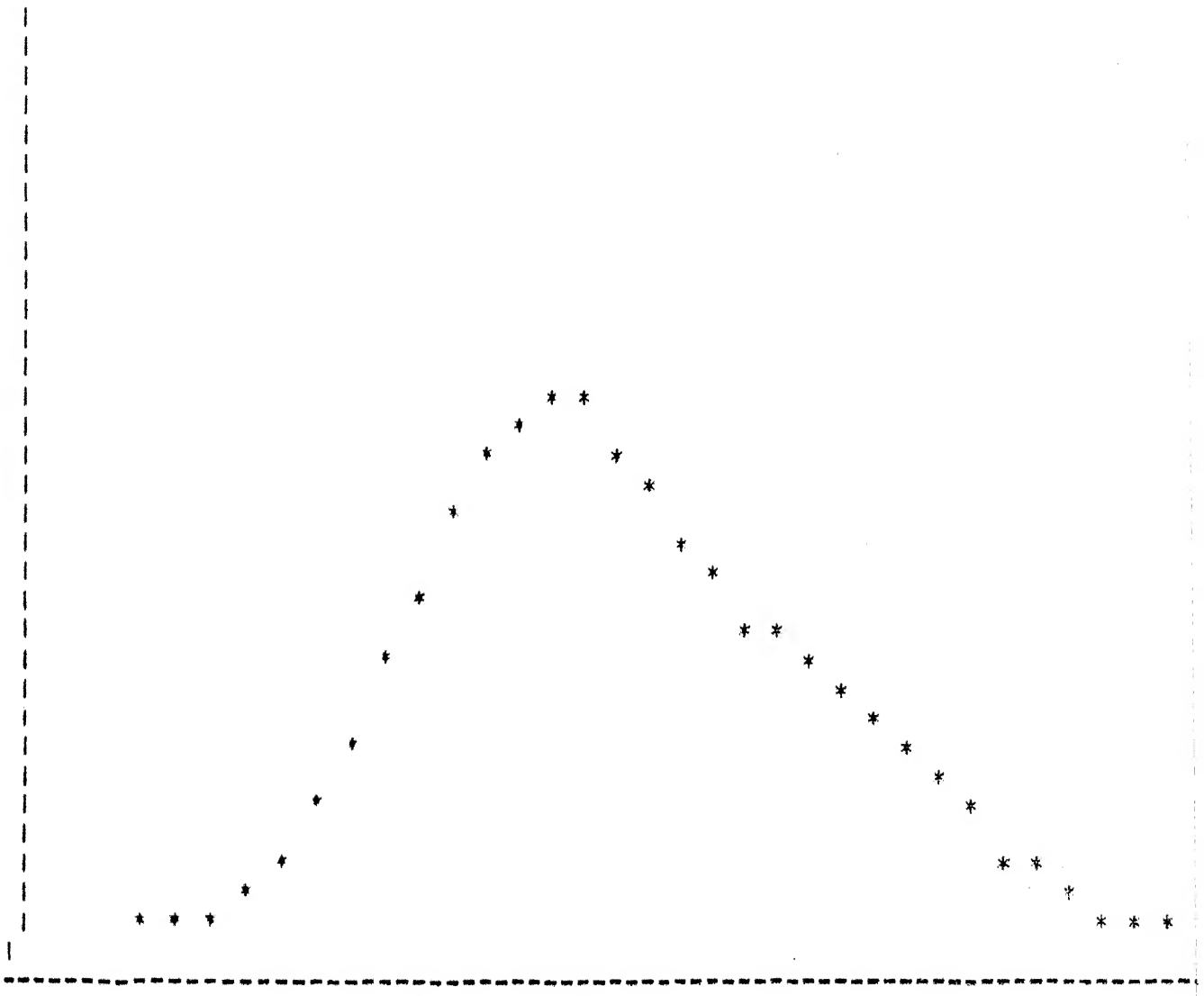




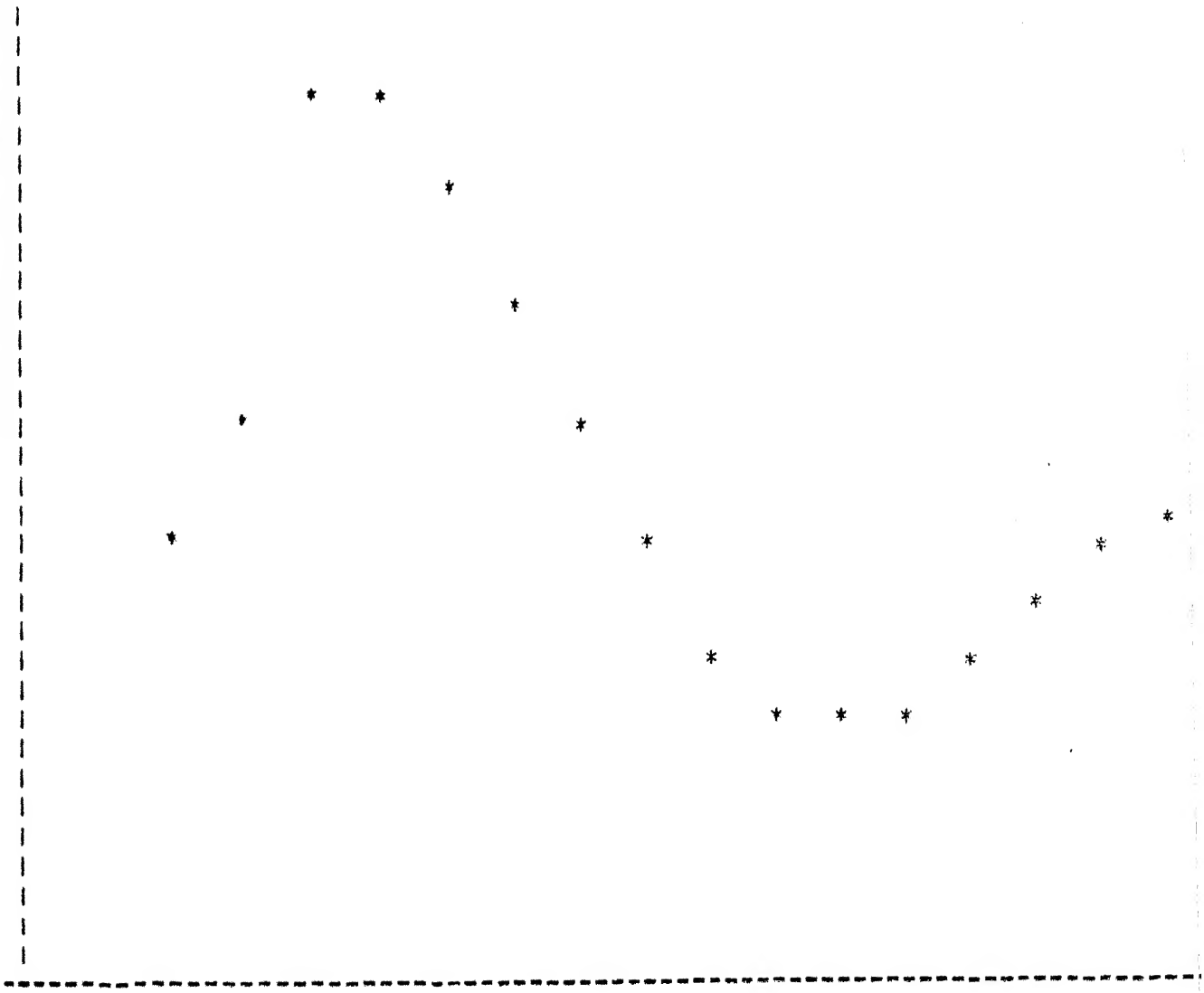
THE OBJECT IMAGE OF SIZE 16



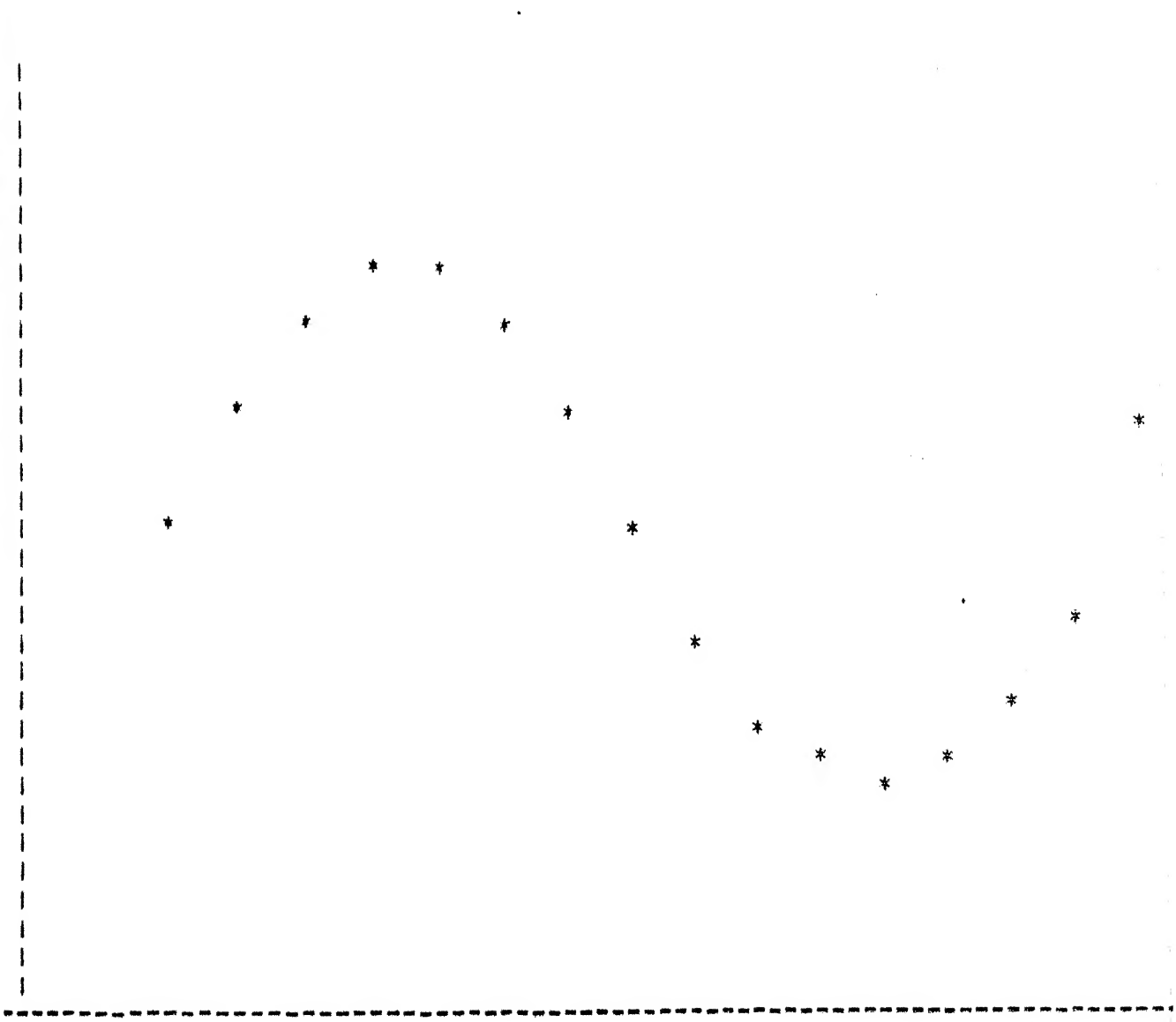
DEGRADED IMAGE WITH NOISE N(0,0.2)



THE DISTORTED IMAGE



RECOVERED IMAGE USING FRIEDEN'S ALGORITHM



```
C     ------------------------------------------------------------------
C     PURPOSE: TO FIND MAXIMUM LIKELIHOOD ESTIMATES FOR THE STEP PISTON
C
C     DATA  AND THE MAIN PROGRAM ARE WRITTEN BY CHARLES WHITE AS
C     SUBMITTED IN THE COMPUTATIONAL DATA LAND FOR A CHAPTER ENTITLED AN
C     EXTRA FILE USE IN THE PROGRAM FOR ESTIMATION OF THE PISTON.
C     ------------------------------------------------------------------
      DIMENSION DAT(N,M),A(T,2),
     1 A(2,2),A(2,1),XTX(2,2),XTY(2,1),YTY(1,1),TMP(1,1)
      OPEN(UNIT=1,FILE='PISTON.DAT',STATUS='OLD')
      READ(1,*) ((DAT(I,J),J=1,M),I=1,N)
      CLOSE(1)
      DO 7 I=1,N
      X(I,1)=1.0
      X(I,2)=DAT(I,1)
      Y(I,1)=DAT(I,2)
      CONTINUE
      CALL MATMUL(X,N,2,X,2,XTX)
      CALL MATMUL(X,N,2,Y,1,XTY)
      CALL MATMUL(Y,1,N,Y,1,YTY)
      WRITE(6,*) 'XTX'
      WRITE(6,*) ((XTX(I,J),J=1,2),I=1,2)
      WRITE(6,*) 'XTY'
      WRITE(6,*) ((XTY(I,J),J=1,1),I=1,2)
      WRITE(6,*) 'YTY'
      WRITE(6,*) ((YTY(I,J),J=1,1),I=1,1)
      CALL INVERT(XTX,2,2,A,IER)
      WRITE(6,*) 'IER=',IER
      IF(IER.EQ.1) GO TO 99
      CALL MATMUL(A,2,2,XTY,1,B)
      WRITE(6,*) 'B'
      WRITE(6,*) ((B(I,J),J=1,1),I=1,2)
      SSE=YTY(1,1)-B(1,1)*XTY(1,1)-B(2,1)*XTY(2,1)
      WRITE(6,*) 'SSE=',SSE
      MSE=SSE/(N-2)
      WRITE(6,*) 'MSE=',MSE
      DO 10 I=1,2
      DO 10 J=1,2
      A(I,J)=MSE*A(I,J)
      CONTINUE
      WRITE(6,*) 'COVARIANCE MATRIX'
      WRITE(6,*) ((A(I,J),J=1,2),I=1,2)
      SE1=SQRT(A(1,1))
      SE2=SQRT(A(2,2))
      WRITE(6,*) 'SE1=',SE1,'SE2=',SE2
      T1=B(1,1)/SE1
      T2=B(2,1)/SE2
      WRITE(6,*) 'T1=',T1,'T2=',T2
      DF=N-2
      CALL PROBT(T1,DF,P1,IER)
      CALL PROBT(T2,DF,P2,IER)
      WRITE(6,*) 'P1=',P1,'P2=',P2
      STOP
7     FORMAT(1X,'X=',2F10.3)
8     FORMAT(1X,2F10.4)
10    FORMAT(1X)
      WRITE(6,*)
      DO 11 I=1,N
      WRITE(6,*) (X(I,J),J=1,2),Y(I,1)
11    CONTINUE
      WRITE(6,*)
      WRITE(6,*) 'MEANS'
      DO 13 I=1,2
      XM(I)=XM(I)/RN
      WRITE(6,*) XM(I)
13    CONTINUE
      WRITE(6,*)
      TYPE 3
3     FORMAT(1X,'STANDARD DEVIATIONS')
      DO 20 I=1,2
      SD(I)=SQRT(SD(I)/(RN-1.0))
      WRITE(6,*) SD(I)
20    CONTINUE
      WRITE(6,*)
      DO 30 I=1,2
      DO 30 J=1,2
      XM(I)=XM(I)-XM(J)*A(I,J)
30    CONTINUE
      WRITE(6,*) XM(1)
      WRITE(6,*) XM(2)
      STOP
99    WRITE(6,*) 'ERROR'
      STOP
      END
```

```
      DO 60 I=1,4
      DO 70 J=1,4
      DRT(I,J)=RP(I,J)
70    CONTINUE
60    CONTINUE
      TYPE 5
5     FORMAT(1X,'PKT OVER')
      DO 90 I=1,N
      DO 90 J=1,N
      SUM=0.0
      DO 100 K=1,4
      SUM=SUM+RG(I,K)*RP(K,J)
100   CONTINUE
      X(I,J)=SUM
90    CONTINUE
80    CONTINUE
      TYPE 6
6     FORMAT(1X,'Y OVER')
      AMAX=0.0
      DO 120 I=1,N
      DO 125 J=1,N
      SUM=0.0
      DO 130 K=1,N
      SUM=SUM+X(I,K)*DRT(K,J)
130   CONTINUE
      IMG(I,J)=SUM
      AMAX=AMAX+SUM
125   CONTINUE
120   CONTINUE
      WRITE(41,*) ((IMG(I,J),J=1,N),I=1,N)
C     INITIALIZATION OF THE CONVOLUTION MATRIX [A]
      DO 140 I=1,250
      M1=1
      CALL RDMAT(MT,M1,M2,M3)
      DO 1150 I=1,64
      A(I,1)=AMAT(I)
1150  CONTINUE
      AINT(1,1)=M1
      AINT(1,2)=M2
      AINT(1,3)=M3
140   CONTINUE
      DO 160 I=1,NX
      DO 160 J=1,NX
      SUM=0.0
      DO 170 K=1,NX
      SUM=SUM+S(K,I)*S(K,J)
170   CONTINUE
      ATA(I,J)=SUM/SIGMA
160   CONTINUE
      WRITE(43,*),((A(I,J),J=1,NX),I=1,64)
      WRITE(44,*),((AINT(I,J),J=1,3),I=1,NX)
      WRITE(40,*),((ATA(I,J),J=1,NX),I=1,NX)
      STOP
      END
```

```
C     ------------------------------------------------------------------
      SUBROUTINE [illegible]
      [illegible]
      [illegible]
      [illegible]
      DO [illegible]
      SK(N,1)=0.0
400   CONTINUE
      M1=(M-1)/16
      M2=M-16*M1
      M=N+1
      IF(N.EQ.0) GOTO 410
      IF(N.EQ.1) GOTO 430
      IF(N.EQ.2) GOTO 450
      M=N-(N-1)
      Y=M-1
      GOTO 440
410   M=0
      Y=0
      GOTO 460
430   M=N-1
      Y=0
440   IF(N2.EQ.0)GOTO 420
      IF(N2.EQ.1)GOTO 450
      IF(N2.EQ.2)GOTO 460
      M2=0-(M2-1)
      M=0*A+1-M
      Y=M2-1
      GOTO 470
450   M2=0
      M=0*A+1
      Y=0
      GOTO 470
460   M2=N-2
      M=0*A+1
      Y=0
470   I=M-1
      L1=A+1
      L2=L1+M1-1
      L3=I+1
      L4=L3+M2-1
      DO 500 K=L1,L2
      DO 510 L=L3,L4
      SK(N,J)=DK(K,1)*DK(K,L)
      J=J+1
510   CONTINUE
      J=J+R-M2
500   CONTINUE
      RETURN
      END
C     ------------------------------------------------------------------
C     ------------------------------------------------------------------
```

```
C     ------------------------------------------------------------------
C     [illegible] AND [illegible] [illegible] [illegible] [illegible]
C     ------------------------------------------------------------------
      [illegible]
      [illegible]
      I=10
      [illegible]
      SIGMA=0.5
      [illegible]
      [illegible]
      [illegible]
      [illegible]
      DO [illegible]
      DO [illegible]
      [illegible]
      [illegible]
      [illegible]
40    [illegible]
30    [illegible]
      [illegible]
      [illegible]
      [illegible]
      [illegible]
      F.0
C     ------------------------------------------------------------------
```

```
C     ------------------------------------------------------------------
C     MAXIMUM ENTROPY FORMULATION FOR Z=0 LOGGA VS DPL DISCRIMINANT
C     SOLUTION ALGORITHM BY SOLVING A SYSTEM OF INITIAL VALUE DIFFERENTIAL
C     EQUATIONS.
C     ------------------------------------------------------------------
      DIMENSION A(256,64),P(256),B(64), FF(64),AT(64,256),AIA(64,64)
      DIMENSION X(64),XA(64),AMAT(64,2) ,ATMG(16,16),AIST(256,4)
      DIMENSION PIC(64),IOG(N,5)
      REAL LAM,LAMO,MU,LOGL,LPS,MF,SIGMA,SUM,AA,WW,C,T
      INTEGER I,J,K,IC,PIC,WX,WA,IWO,ATMT
      COMMON/AIRG/AIRG,N,P,FF,ATM,WX,AIDA
      COMMON/VAR/MF,LAMO,LAMD
      COMMON/CONST/C,WW,MU,WA,IX,IX
      N=64
      M=256
      K=N+1
      WX=N*2
      WA=M*2
      SIGMA=0.01
      EPS=0.1
      OPEN(UNIT=22,DEVICE='DSK',FILE='FOR02.DAT')
      OPEN(UNIT=23,DEVICE='DSK',FILE='FOR03.DAT')
      OPEN(UNIT=25,DEVICE='DSK',FILE='FOR04.DAT')
      OPEN(UNIT=24,DEVICE='DSK',FILE='FOR06.DAT')
      READ(22,9) ((A(I,J),J=1,WA),I=1,WX)
      READ(23,9) ((ATMG(I,J),J=1,N),I=1,N)
      READ(24,9) ((ATAT(I,J),J=1,WA),I=1,WX)
      READ(25,9) ((AIST(I,J),J=1,3),I=1,WA)
      K=1
      DO 660 J=1,N
      DO 660 I=1,M
      A(K)=ATMG(I,J)
      K=K+1
660   CONTINUE
      T=0.0
      DO 5 I=1,WX
      T=T+B(I)
5     CONTINUE
      DO 25 I=1,WA
      XA(I)=0.0
      DO 26 I=1,WX
      AT(I,J)=A(J,I)
      XA(I)=XA(I)+AT(I,J)*B(J)
26    CONTINUE
      XA(I)=XA(I)/SIGMA
25    CONTINUE
      ACCEPT*,MAIT,INCP,MU
      DO 31 I=1,WA
      F(I)=EXP(MU-1.0)
30    FORMAT(
      KK=0
      LAMO=0.0
333   ACCEPT*,MAIT,INCP
77    CALL CALDF
```

```
      IF(NSG(JF),LE,ESS)GOTO 40
      IF(JF.GT.NA)GO TO 31
      LA=LA+ALT(JF)
      GO TO 32
31    LA=LA+ALT(NA)
32    CONTINUE
      NA=NF+1
      IF(F.GT.FMIT)GOTO 999
      CALL ORDER
      DO 10 I=1,NA
      F(I)=FF(I)
40    CONTINUE
      LA=LA+1
      GOTO 77
99    CONTINUE
      SUM=0.0
      DO 50 I=1,NA
      SUM=SUM+AF(I)
      PIC(I)=IFIX(F(I)*1.0)
50    CONTINUE
      DO 77 I=1,NA
      DO 77 J=1,NA
      K=AF(J-1)+1
      IMG(I,J)=PIC(K)
770   CONTINUE
      WRITE(10,998) ((IMG(I,J),J=1,NA),I=1,NA)
998   FORMAT(1X,915,/)
      GOTO 55
999   TYPE 111,NF,SUM,LA,NA
111   FORMAT(1X,'NO OF ATTRACTORS EXCEEDS',16,3F15.6)
      DO 202 I=1,NA
      F(I)=FF(I)
202   CONTINUE
      GOTO 333
55    STOP
      END
C     ------------------------------------------------------------------
      SUBROUTINE CALOP
      DIMENSION P(256,64),D(256),F(64),FF(64),AF(256),ALT(256,3)
      REAL LAM,IMAG,SUM,A,DF,SUM,XX
      INTEGER I,J,NX,NX,NINT
      COMMON/ARRAYS/A,D,F,FF,ATA(64,64),XA(64),NINT
      COMMON/VAR/F,LAM,LA
      COMMON/CONST/N,M,L,SUM,A,NX,NA
      NF=0.0
      DO 10 I=1,NA
      SUM=0.0
      J=ALT(I,3)
      DO 20 K=1,ALT(I,1)
      DO 25 II=1,ATA(I,2)
      SUM=SUM+A(I,J)*F(J)
      J=J+1
25    CONTINUE
      J=J+N-ALT(I,2)
```

```
20          C[illegible]E
            AF(A)=SUM-P(I)
            DF=DF+F(I)*A(I)
10          CONTINUE
            TR=1.0*((DF/SIG(N))-DX)
            TR=DF-DX
555         PRINT*(I,'THE VALUE OF DF IS:')
            TYPE*,DF
            RETURN
            END
C           ------------------------------------------------------------
            SUBROUTINE SOLVE
            DIMENSION P(64,64),B(64),FK(64,64),A(256,64),AINV(256,2)
            REAL LAMDA,LAMO,SIGMA,DF,A,B,F,FF,ATA,AX
            INTEGER I,J,K,N,NX,NY,NN
            COMMON/APPROX/A,P(256),B(64),FF(64),ATA(64,64),AX(64),NX,
            COMMON/DATA/F,LAMO,LAMDA
            COMMON/CONST/I,J,K,N,STA,NN,NX,NA
            DO 10 I=1,NA
            DO 10 J=1,NA
            P(I,J)=0.0
10          CONTINUE
            DO 20 I=1,NX
            FK(I,I)=1.0/F(I)
20          CONTINUE
            DO 30 I=1,NA
            DO 30 J=1,NA
            P(I,J)=FF(I,J)+LAMDA*ATA(I,J)
30          CONTINUE
            DO 40 I=1,NA
            SUM=0.0
            DO 50 J=1,NA
            SUM=SUM+ATA(I,J)*F(J)
50          CONTINUE
            B(I)=SUM*(2*LAMO-LAMDA)+1.0
70          CONTINUE
            DO 60 I=1,NX
            B(I)=B(I)+AX(I)*(LAMDA-LAMO)
60          CONTINUE
            TR=0.0
            DO 110 J=2,NX
            TR=TR+P(1,J)*F(J)
110         CONTINUE
            FK(1)=(B(1)-TR)/P(1,1)
            DO 120 I=2,NA
            TR=0.0
            DO 130 J=1,I-1
            TR=TR+P(I,J)*FF(J)
130         CONTINUE
            FF(I)=B(I)-TR
            IF(I.LT.NA)GOTO 139
            SUM=0.0
            DO 140 J=I+1,NX
            SUM=SUM+P(I,J)*F(J)
```

```
100       [illegible]
          Fr(i)=Fr(i)-Sy
101       Fr(i)=Fr(i)/F(1,1)
120       [illegible]
          [illegible]
          [illegible]
C         ************************************************************
C         ************************************************************
```

...(1,9)

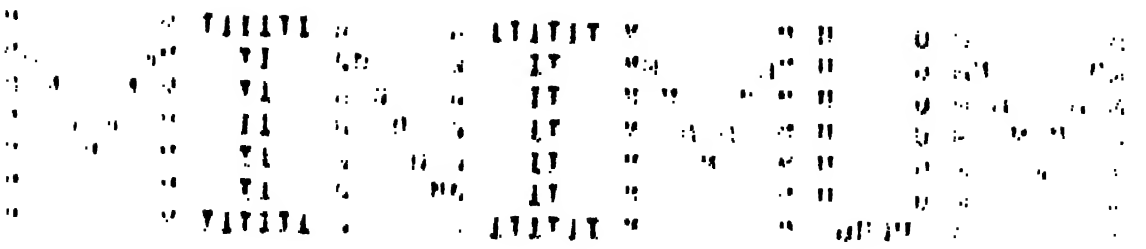
| | | | | | | | |
|---|---|---|---|---|---|---|---|
| 1 | 5 | 7 | 12 | 12 | 9 | 5 | 1 |
| 2 | 7 | 14 | 18 | 18 | 14 | 8 | 4 |
| 3 | 14 | 18 | 19 | 19 | 18 | 12 | 7 |
| 4 | 18 | 21 | 21 | 21 | 19 | 15 | 12 |
| 5 | 19 | 19 | 21 | 21 | 19 | 18 | 12 |
| 6 | 15 | 15 | 18 | 19 | 18 | 12 | 7 |
| 7 | 7 | 12 | 15 | 15 | 12 | 8 | 5 |
| 8 | 1 | 7 | 12 | 12 | 9 | 5 | 1 |

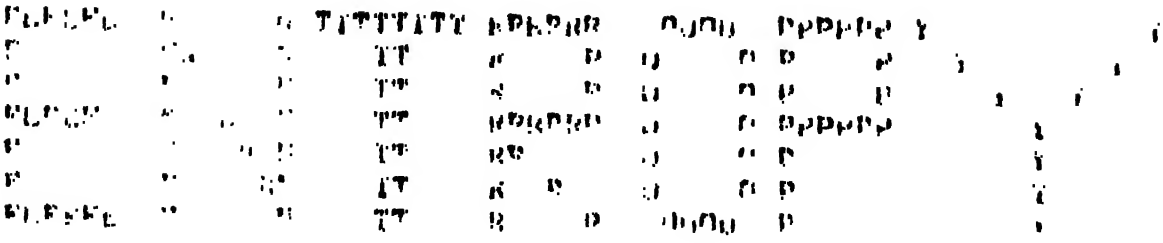
...(4,9) ... ...(1,9) ... ...

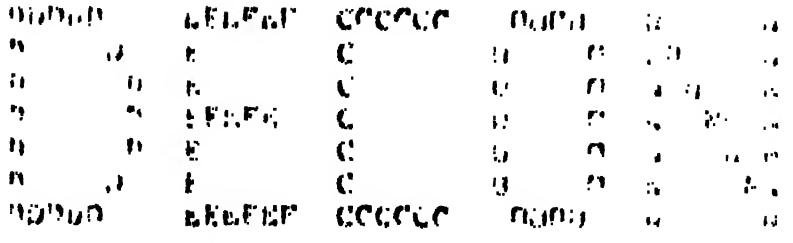
| | | | | | | | |
|---|---|---|---|---|---|---|---|
| 2 | 4 | 7 | 14 | 12 | 7 | 4 | 2 |
| 5 | 8 | 12 | 15 | 16 | 13 | 8 | 5 |
| 9 | 13 | 15 | 17 | 18 | 17 | 13 | 9 |
| 11 | 15 | 16 | 20 | 20 | 19 | 15 | 11 |
| 11 | 16 | 19 | 21 | 21 | 19 | 16 | 11 |
| 9 | 13 | 17 | 20 | 19 | 16 | 12 | 8 |
| 4 | 9 | 14 | 16 | 15 | 12 | 8 | 5 |
| 1 | 5 | 9 | 11 | 11 | 8 | 5 | 2 |

...(3,9) ... ...
...(1,8)=2.

| | | | | | | | |
|---|---|---|---|---|---|---|---|
| 2 | 5 | 9 | 11 | 10 | 8 | 5 | 4 |
| 4 | 9 | 13 | 15 | 15 | 12 | 8 | 6 |
| 7 | 14 | 18 | 19 | 19 | 16 | 12 | 8 |
| 10 | 17 | 21 | 22 | 21 | 18 | 14 | 10 |
| 10 | 16 | 21 | 22 | 21 | 19 | 14 | 10 |
| 5 | 9 | 13 | 15 | 14 | 12 | 8 | 6 |
| 3 | 6 | 9 | 11 | 10 | 8 | 6 | 4 |





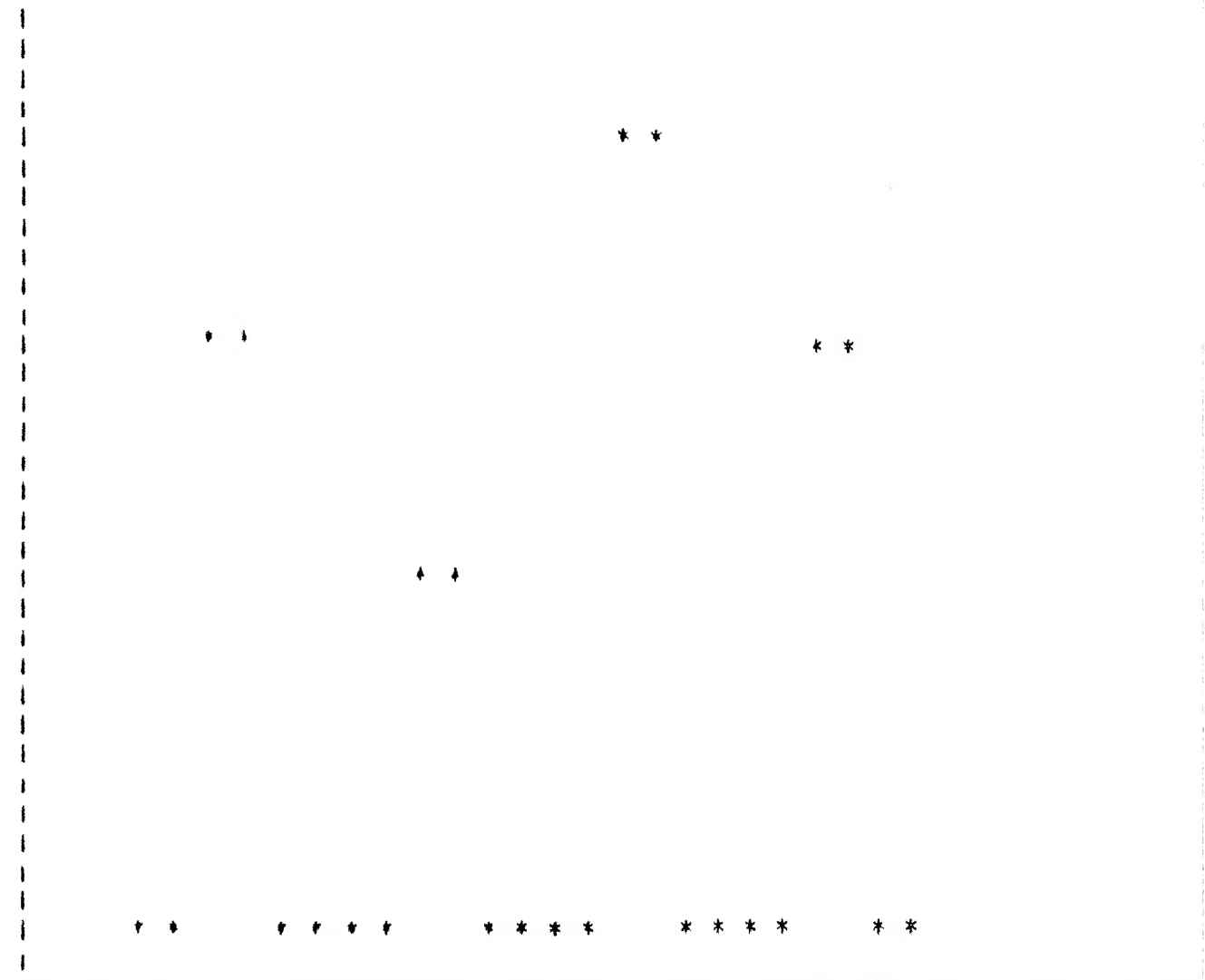


```
C     ------------------------------------------------------------------
C     "IRA'ON" ENTROPY BASED DECONVOLUTION. THIS PROGRAM FIRST A FILTER
C     WHICH DECONVOLVES THE DEGRADED IMAGE DATA BY MINIMIZING THE
C     ENTROPY OF THE ESTIMATED OF THE ESTIMATED OBJECT DATA. THE ONLY
C     DATA AVAILABLE IS THE DEGRADED IMAGE DATA. THE KNOWLEDGE ABOUT
C     THE CONVOLUTION MATRIX OF THE IMAGING SYSTEM IS ASSUMED USING OF
C     THE MAXIMUM ENTROPY METHODS. THE OBJECT DATA IS HOWEVER KNOWN
C     TO BE INFORMATIVE WITH NON-UNIFORM PROBABILITY DISTRIBUTION.
C     ------------------------------------------------------------------
      REAL CONST1,CONST2
      INTEGER NG,NPIED,NRUNS
      REAL CI(1),CU(1),X(34,4)
      REAL ETA,F,RHO,STEPMX,XTOL
      INTEGER I,ICONTD,IFAIL,IPRINT,ICHO,IT,IB,P,MAXCAL,NOUT
     1,NX,P,NS,NA
      LOGICAL BASET
      REAL G(35),HEAD(35),IMG(40),X(3),A1(3),AU(3),..(5000)
      INTEGER IW(200),OUT(100)
      EXTERNAL E04ABY
      COMMON/NOUT/NOUT
      COMMON/CONSTS/CONST1,CONST2
      COMMON/AI/AEQ,NEQ,IRUGE,CI,CU
      DATA NOUT/50/
      NG=32
      P=1
      NM=NG*P=1
      OPEN(UNIT=23,DEVICE='DSK',FILE='FOR35.DAT')
      READ(23,*) (IMG(I),I=1,NG)
      CONST1=0.0
      DO 10 I=1,NG
      CONST1=CONST1-IMG(I)
10    CONTINUE
      CONST2=-1.0
      DO 20 I=1,NM
      DO 20 J=1,P
      Y(I,J)=0.0
20    CONTINUE
      DO 30 I=1,P
      DO 30 J=1,NG
      Y(I+J-1,I)=IMG(J)
30    CONTINUE
      N=P
      NEQ=1
      NINEQ=NM
      NRUGE=0
      NA=NEQ+NINEQ+NRNGE
      IPRINT=0
      NX=N+MINEQ+NRNGE
      MAXCAL=100*(N+5)*NX
      ETA=0.50
      XTOL=100.0*SQRT(X02AAF(ATOL))
      STEPMX=10000.0
      ICHO=1
      IBOUND=0
```

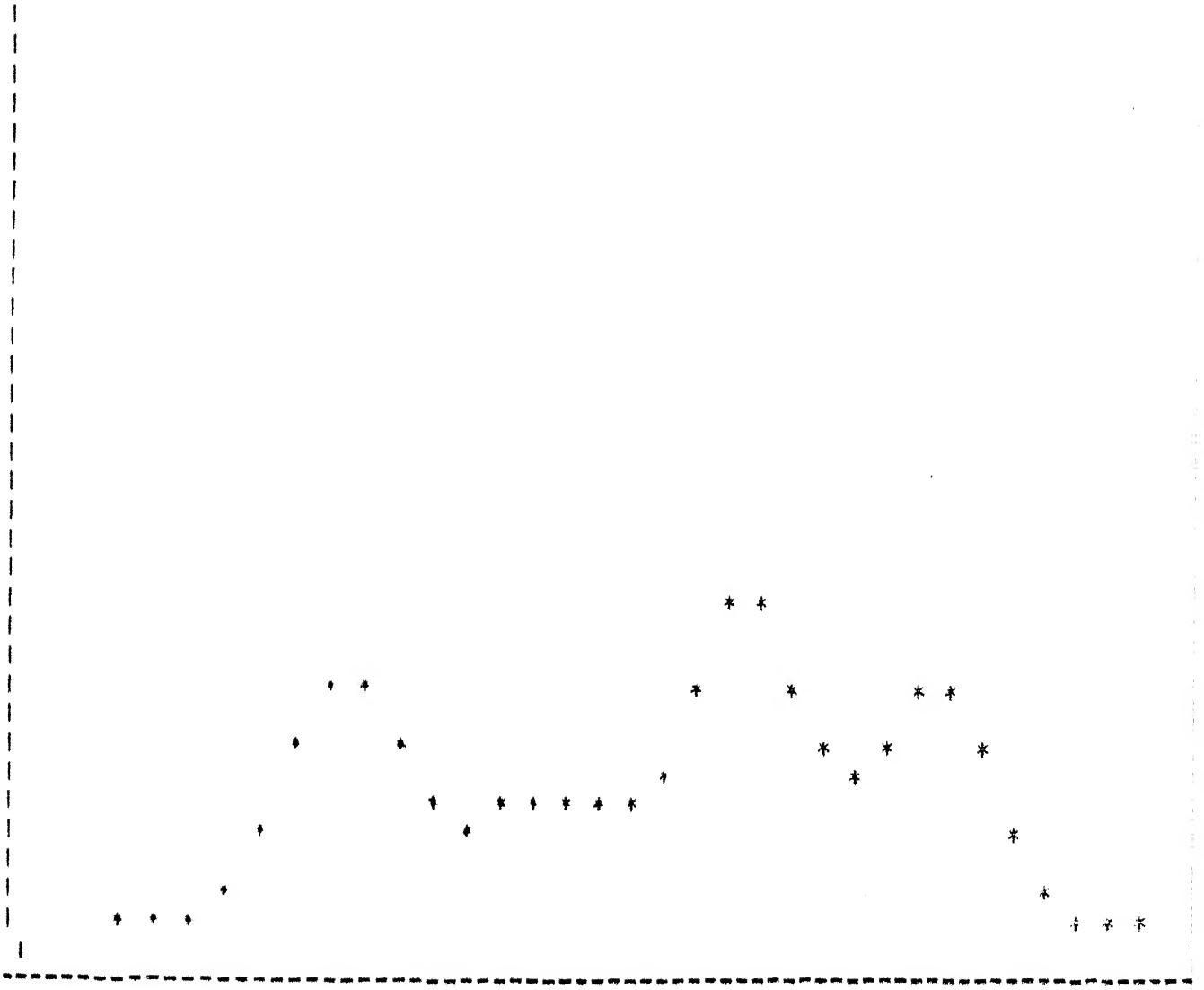
```
      XL(I)=0.0005
      XU(I)=1.0E+6
      DO 33 I=2,7
      XL(I)=-1.0E+6
      XU(I)=1.0E+6
33    CONTINUE
      EASOLN=.FALSE.
      DO 40 I=1,M
      X(I)=10.0
40    CONTINUE
      NN(1)=1.0
      WRITE(*,*)'ENTER 0 FOR SQUARE,1 FOR CIRCLE,2 FOR ELLIPSE'
      READ(*,*)IPATH
      IF(IPATH.EQ.1) STOP
      CALL EQUARE(N,MM,NN,NRANGE,M,EQUAY,IPRINT,EASOLN,ETA,
     1 XU,STEPX,CL,CU,ICL,IRULE,XL,AU,IASSET,X,RHO,RLAM,
     2,F,T,IER,IW,W,IPATH)
      WRITE(*,*) IPATH
      IF(IPATH.EQ.1) STOP
      WRITE(NOUT,997) F
      IF(IPATH.NE.2)WRITE(NOUT,996) (X(I),I=1,N)
      IF(IPATH.EQ.2)WRITE(NOUT,995) (X(I),I=1,N)
      WRITE(NOUT,994)
      WRITE(NOUT,993) (I,C(I),I=1,M)
      WRITE(NOUT,992) RHO
      WRITE(NOUT,991) (RLAM(I),I=1,M)
      SUMR=0.0
      DO 50 J=1,MM
      SUM=0.0
      IF(J.GT.0)GOTO 5
      JJ=J
      GOTO 6
5     JJ=0
6     CONTINUE
      IF(J.GT.NN)GOTO 7
      KK=1
      GOTO 8
7     KK=J-NN+1
8     CONTINUE
      DO 60 I=KK,J
      SUM=SUM+X(I)*A(J,I)
60    CONTINUE
      OBJ(J)=IFIX(SUM+0.5)
      SUMR=SUMR+SUM
50    CONTINUE
      WRITE(55,99) (OBJ(J),J=1,MM),SUMR
99    FORMAT(///'THE RECONSTRUCTED IMAGE IS'/3415/F12.6)
200   STOP
999   FORMAT(///'MAX ENTROPY BLIND DECON'/)
998   FORMAT(///'ERROR EXIT TYPE',I3)
997   FORMAT(///'FUNCTION VALUE AT EXIT IS',F12.4)
996   FORMAT(1X,'AT THE POINT',3F9.4)
995   FORMAT(1X,'AT THE POINT',3F9.4)
```

```
991   FORMAT(1X,'CORRESPONDING CONSTRAINT VALUE ARE')
993   FORMAT(1X,'C(',I2,')=',F9.4)
992   FORMAT(1X,'IF DEVIATING FROM THE FINAL POINT GIVEN IN A'/
     1'SET MIN TO',F15.4,'A% THE ELEMENTS OF ROW 'I2)
991   FORMAT(1H ,2F9.4)
      STOP
      END
C     ------------------------------------------------------------------
C     SUBROUTINE FUNCT1 COMPUTES THE VALUE OF THE OBJECTIVE FUNCTION
C     AFTER EACH ITERATION.
C     ------------------------------------------------------------------
      SUBROUTINE FUNCT1(IFLAG,N,XC,FC)
      REAL FC,SUM,TERM
      INTEGER IFLAG,N,NM,MM,P
      REAL XC(1),Y(34,3)
      COMMON/LOCAL/Y,P,MM,NM
      XN=0.0
      XD=0.0
      DO 10 I=1,NM
      SUM=0.0
      IF(J.GT.P)GOTO 5
      IJ=J
      GOTO 6
5     IJ=P
6     CONTINUE
      IF(J.GT.NM)GOTO 7
      KK=1
      GOTO 8
7     KK=J-NM+1
8     CONTINUE
      DO 20 K=KK,IJ
      SUM=SUM+XC(K)*Y(I,K)
20    CONTINUE
      XN=XN+SUM**4
      XD=XD+SUM**2
10    CONTINUE
      FC=XN/(XD**2)
      RETURN
      STOP
      END
C     ------------------------------------------------------------------
C     SUBROUTINE CON1 COMPUTES THE VALUES OF THE CONSTRAINTS AFTER EACH
C     ITERATION.
C     ------------------------------------------------------------------
      SUBROUTINE CON1(IFLAG,N,M,XC,CC)
      INTEGER IFLAG,M,N,NM,MM,P
      REAL CC(M),XC(N),SUM
      REAL CONST1,CONST2,Y(34,3)
      COMMON/CONSTS/CONST1,CONST2
      COMMON/LOCAL/Y,P,NM,MM
      SUM=0.0
      DO 10 I=1,MM
      IF(J.GT.P)GOTO 5
      IJ=J
```

```
      GOTO 6
5     JJ=F
6     CONTINUE
      IF(J.GT.NJ)GOTO 7
      KK=1
      GOTO 8
7     KK=J-NJ+1
8     CONTINUE
      DO 20 K=KK,JJ
      SUM=SUM+XC(K)*Y(J,K)
20    CONTINUE
10    CONTINUE
      CC(I)=SUM+CONST1
      DO 44 I=1,NN
      JJ=J+1
      CC(I,J)=0.0
      IF(J.GT.P)GOTO 11
      J,I=J
      GOTO 12
11    JJ=P
12    CONTINUE
      IF(J.GT.NJ)GOTO 13
      KK=1
      GOTO 14
13    KK=J-NJ+1
14    CONTINUE
      DO 55 K=KK,JJ
      CC(I,J)=CC(IJ)+XC(K)*Y(J,K)
55    CONTINUE
C     CC(I,I)=CC(IJ)-1.0
44    CONTINUE
      RETURN
      END
C     ------------------------------------------------------------------
      SUBROUTINE ANOVA(N,Y,X,F,C,NITER,GSNOR,CONV,POSDEF,KNO,NAME)
      REAL CONV,F,GSNOR,KNO
      INTEGER N,P,NF,NITER
      LOGICAL POSDEF
      REAL C(1),RLAM(N),X(1)
      INTEGER NEQ,NTNEQ,NRNGE
      REAL CL(1),CU(1)
      REAL CHORE,CTEMP,DUMMY
      INTEGER I,J,NOUT,K,S,T
      REAL SQRT
      COMMON/MON/NEQ,NTNEQ,NRNGE,CL,CU
      DATA NOUT/50/
      RETURN
      END
C     ******************************************************************
C     ******************************************************************
```

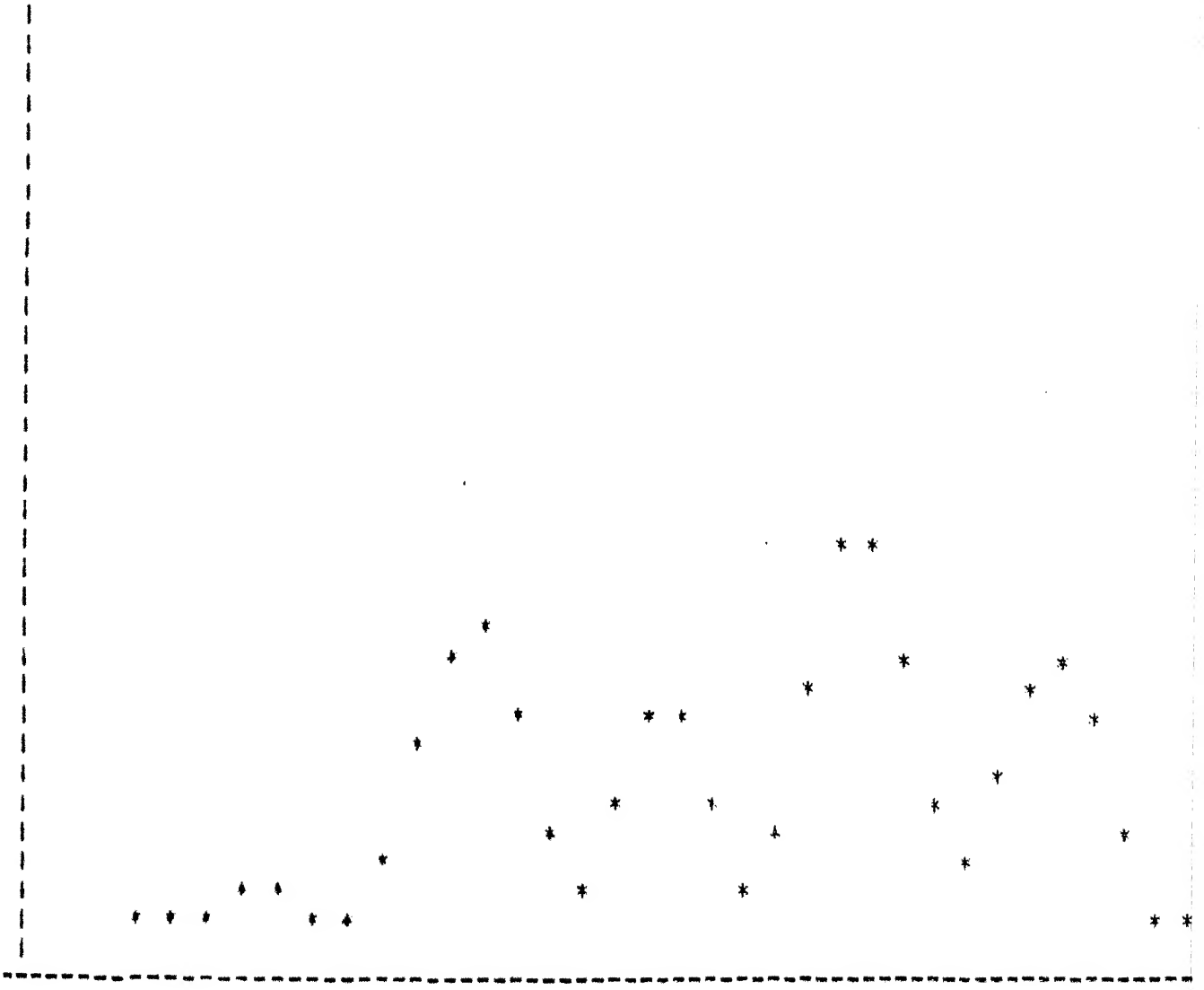


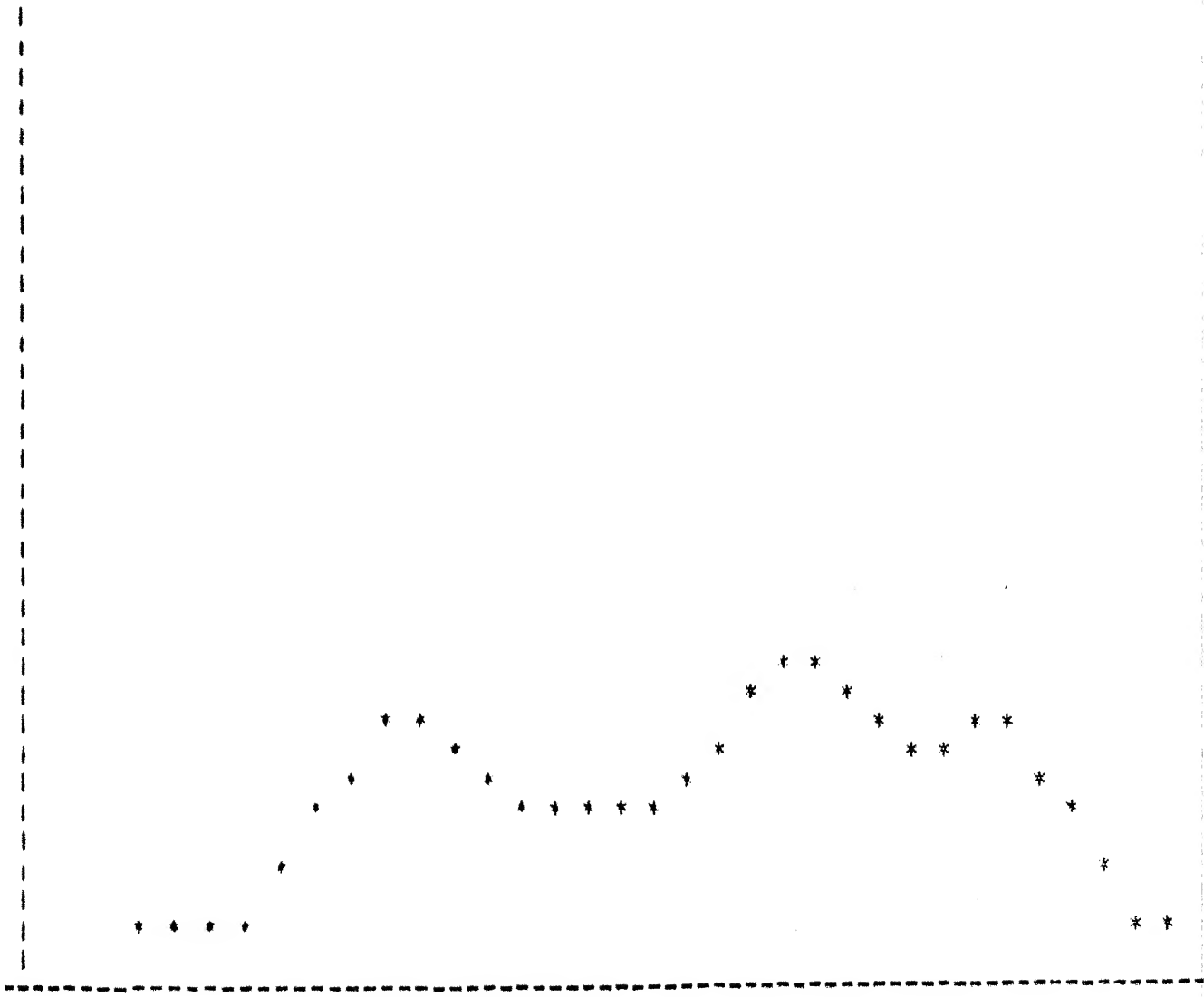
DIFFRACTION LIGHT INTENSITY UNDER 9 DIFFRACTION DIOR FILTER



[illegible] FOUR-WALL ORDER 5 DECOUPLER

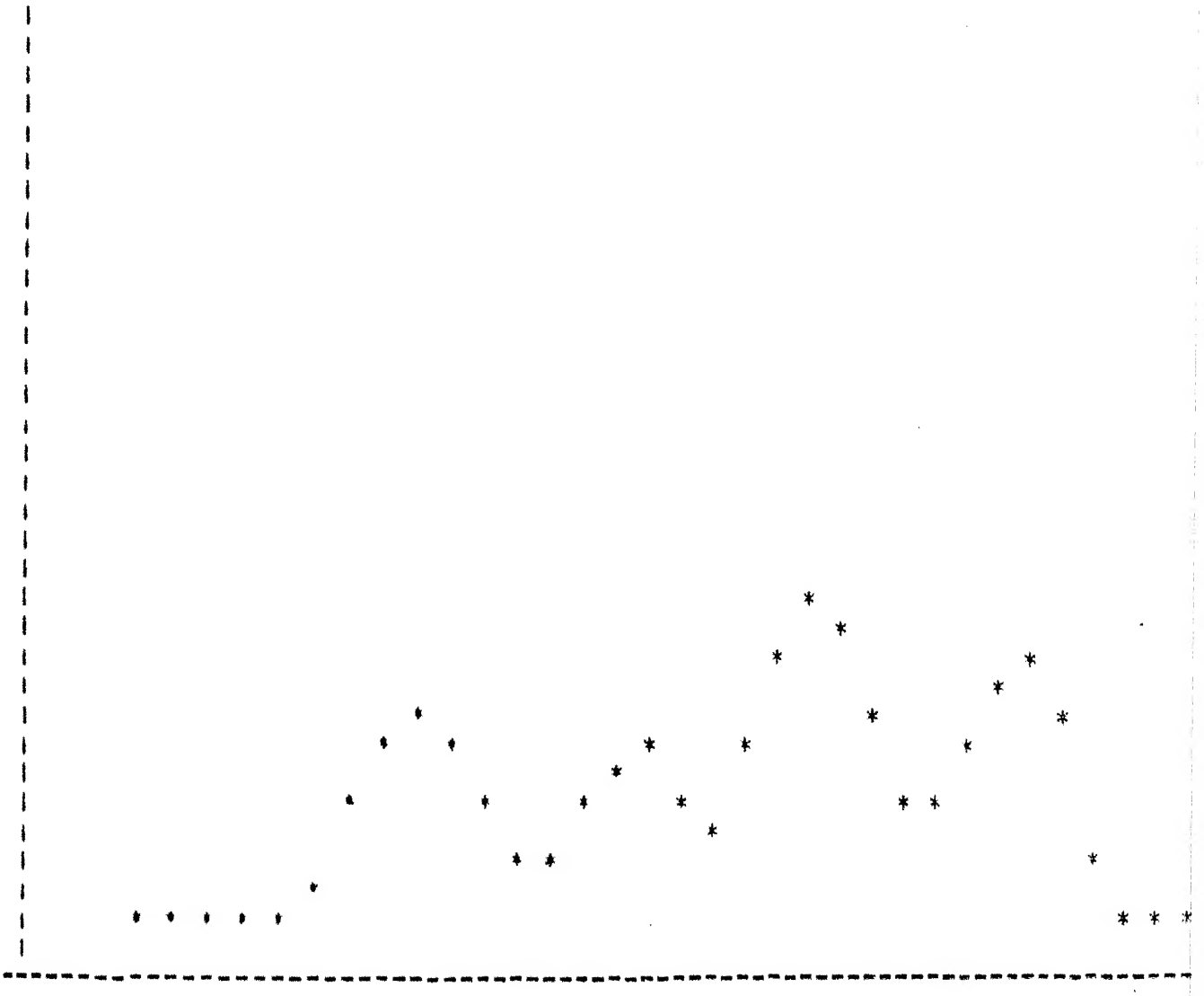
WITH COEFFS: 1.2253 -1.1696 5.0936 -1.0070 0.1777

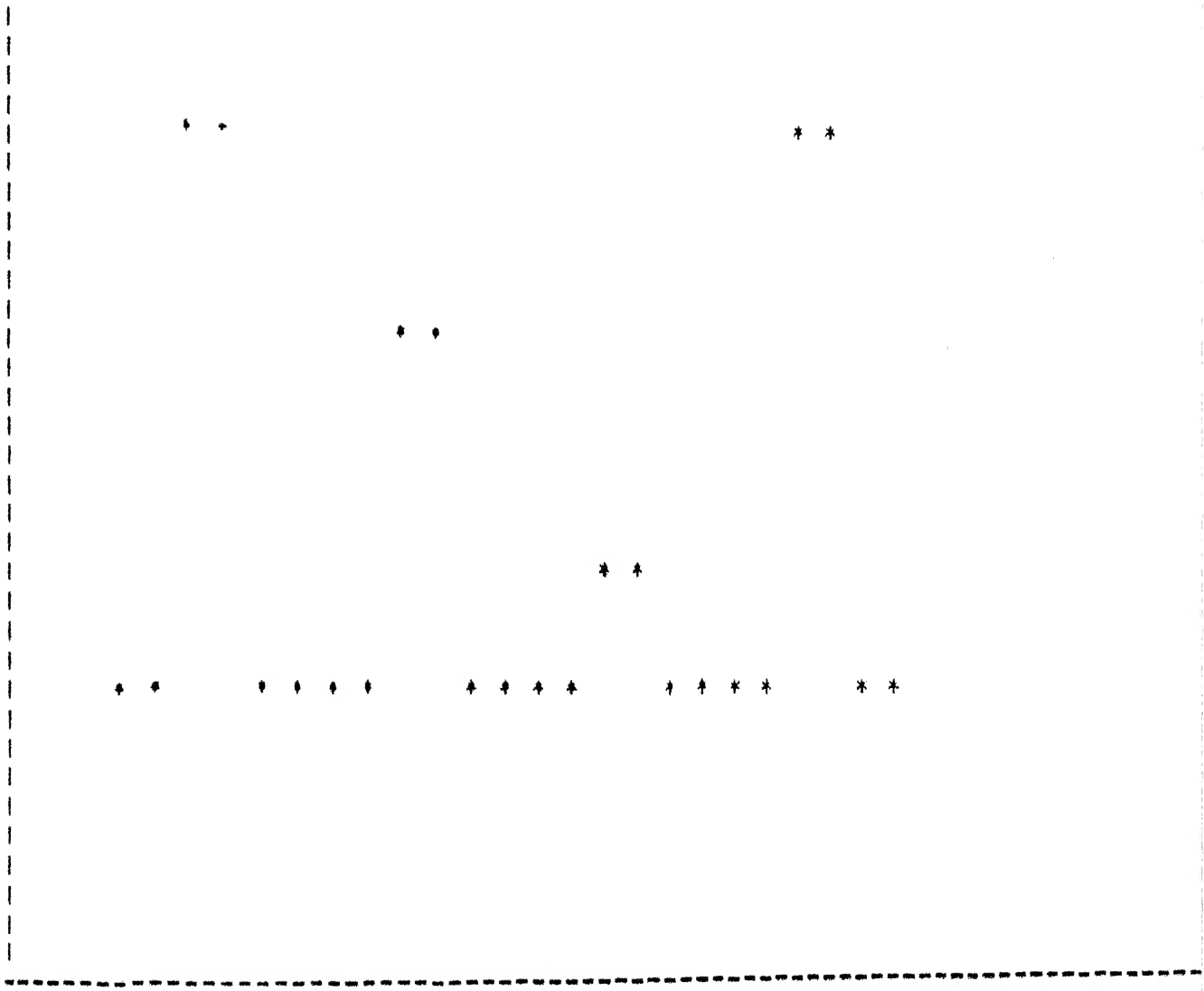




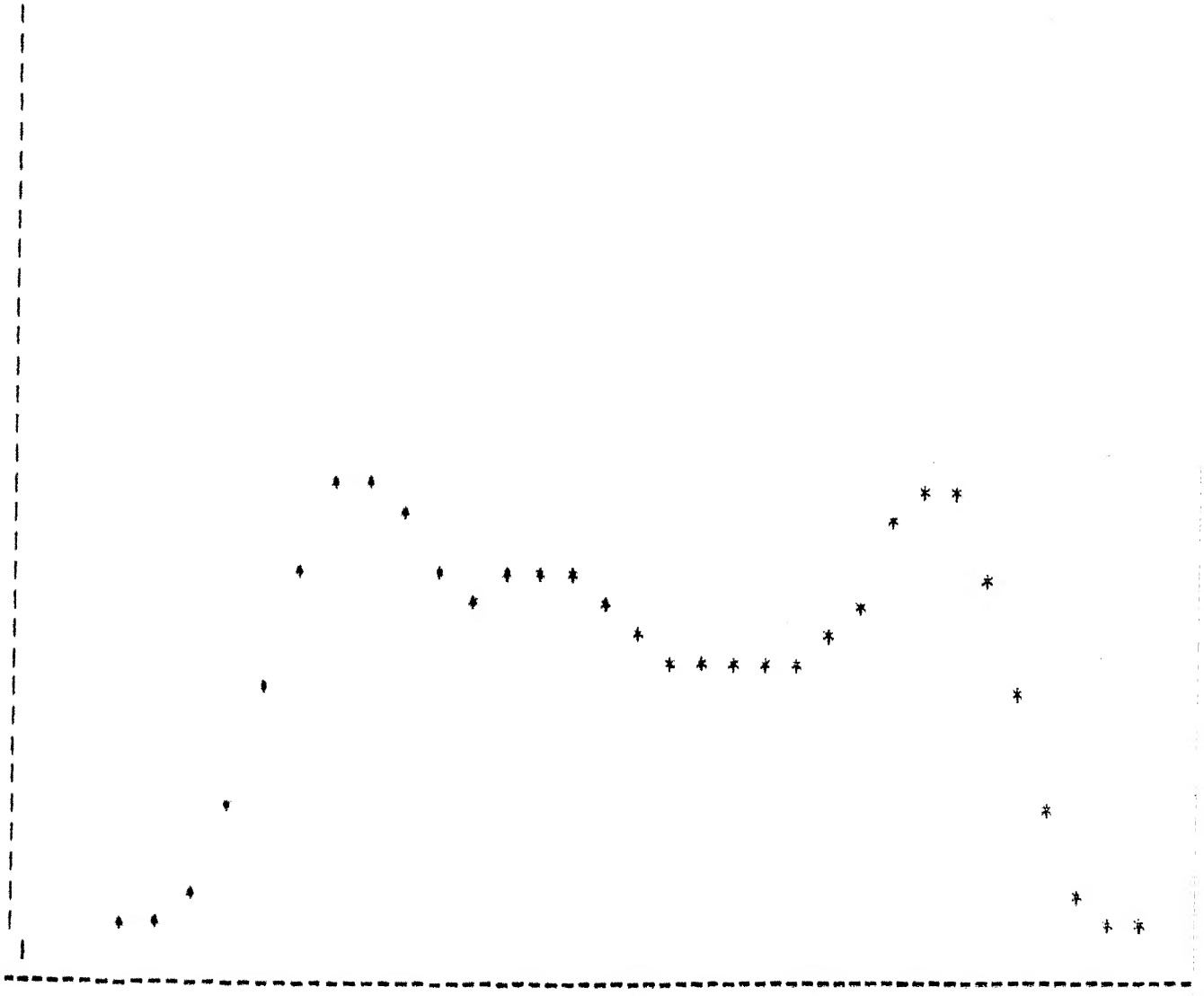
[illegible] ORDER 5 FILTER

[illegible] -8.7051 19.3358 -17.7997 6.2504



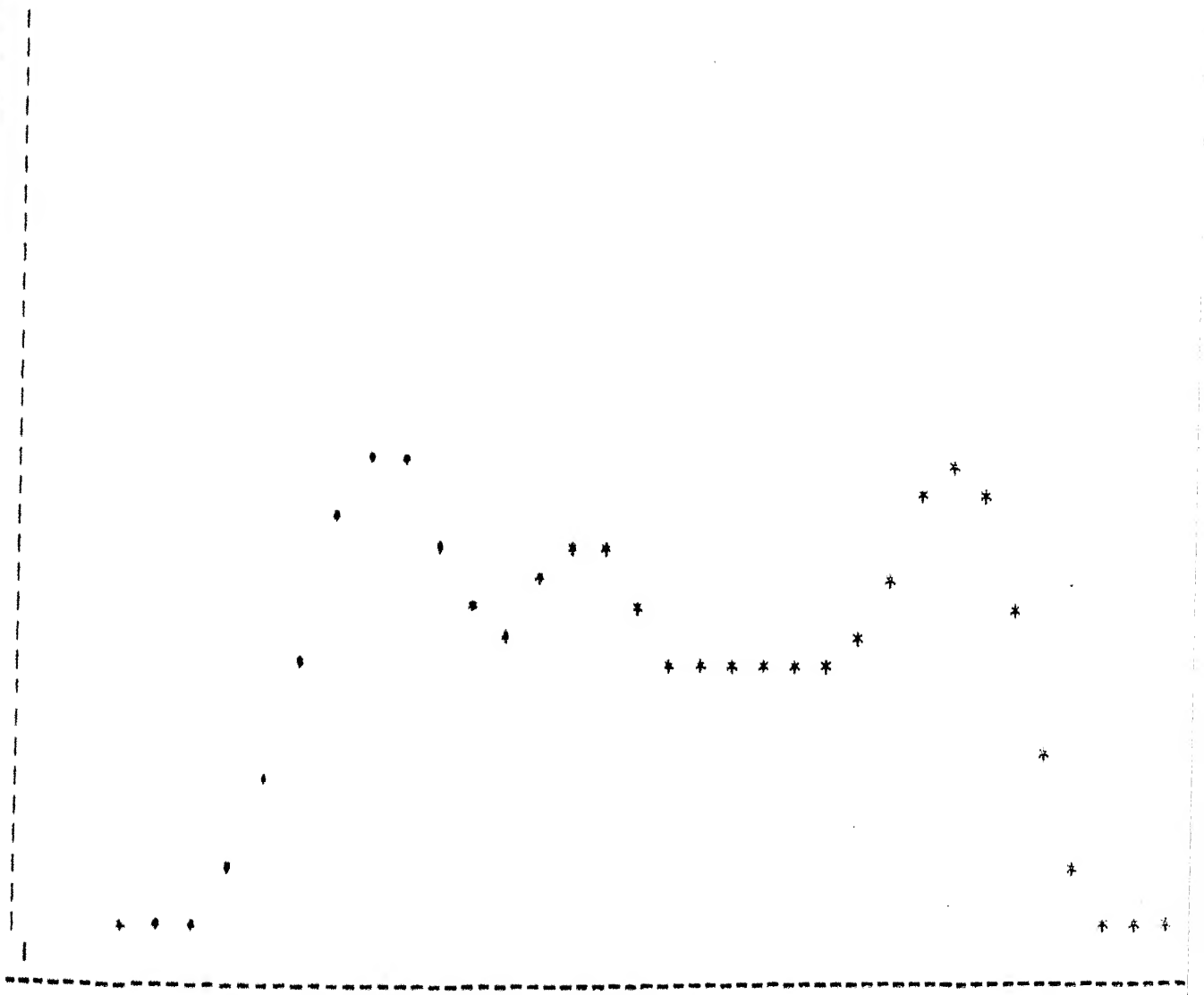


THE DIFFRACTED BLURRED IMAGE WITH ORDER 9 FILTER



[illegible] ORDER 5 DrCOW FILTER

[illegible] 1.1971 0.1749 -0.5459 0.1734



DE RECOVERED IMAGE DATA USING ORDER 3 DECON FILTER

FILTER COEFFS: 0.0008 1.2203 -0.2211